ELSEVIER

Signal Processing 41 (1995) 135–152

SIGNAL PROCESSING

Fast algorithms for analyzing and designing weighted median filters

Ruikang Yang*[,a], Moncef Gabbouj[b], Yrjö Neuvo[c]

[a] *Audio-Visual Signal Processing Lab., Nokia Research Center, Kanslerink 8N, SF-33720, Tampere, Finland*
[b] *Signal Processing Laboratory, Tampere University of Technology, Tampere, Finland*
[c] *Nokia Mobile Phone, Espoo, Finland*

Received 13 September 1993; revised 2 May 1994

Abstract

In this paper, two fast algorithms are developed to compute a set of parameters, called M_i's, of weighted median filters for integer weights and real weights, respectively. The M_i's, which characterize the statistical properties of weighted median filters and are the critical parameters in designing optimal weighted median filters, are defined as the cardinality of the *positive subsets* of weighted median filters. The first algorithm, which is for integer weights, is about four times faster than the existing algorithm. The second algorithm, which applies for real weights, reduces the computational complexity significantly for many applications where the symmetric weight structures are assumed. Applications of these new algorithms include design of optimal weighted filters, computations of the output distributions, the output moments, and the rank selection probabilities, and evaluation of noise attenuation for weighted median filters.

Zusammenfassung

Zwei schnelle Algorithmen werden entwickelt zur Berechnung eines Satzes von Parametern – M_i genannt – für gewichtete Medianfilter mit ganzzahligen und allgemein reellen Gewichten. Die M_i-Werte, welche die statistischen Eigenschaften gewichteter Median-filter charakterisieren und die kritischen Parameter beim Entwurf optimaler gewichteter Median-filter sind, werden definiert als die Mächtigkeit der *positiven Untermengen* gewichteter Medianfilter. Der erste Algorithmus, der mit ganzzahligen Gewichten arbeitet, ist etwa um den Faktor 4 schneller als der existierende Algorithmus. Die zweite Rechenvorschrift (für allgemein reelle Gewichte) verringert den Rechenaufwand für viele Anwendungen erheblich, für welche symmetrische Gewichtsstrukturen angenommen werden. Anwendungen dieser neuen Algorithmen schließen den Entwurf optimaler gewichteter Filter ein, die Berechnung der Ausgangsverteilungen, der Ausgangsmomente und der Rangauswahl-Wahrscheinlichkeiten sowie die Auswertung der Störunterdrückung durch gewichtete Medianfilter.

Résumé

Nous présentons dans cet article deux algorithmes rapides pour le calcul d'un ensemble de paramètres appelés M_i des filtres médians pondérés pour des coefficients de pondération entiers et réels. Les M_i, qui caractérisent les propriétés statistiques des filtres médians pondérés et sont des paramètres critiques pour la conception optimale de ceux-ci, sont

* Corresponding author. Tel.: 358-31-3165679. Fax: 358-31-3165897. E-mail: yang@research.nokia.fi.

SSDI 0165-1684(94)00096-4

définis comme les cardinaux de leurs *sous-ensembles positifs*. Le premier algorithme, qui est conçu pour les coefficients de pondération entiers, est à peu près quatre fois plus rapide que l'algorithme existant. Le second algorithme, qui s'applique aux coefficients de pondération réels, réduit la complexité calculatoire de manière significative dans de nombreuses applications où on suppose des structures de coefficients symétriques. Les applications de ces algorithmes nouveaux incluent la conception de filtres pondérés optimaux, les calculs des distributions de sortie, des moments de sortie, et les probabilités de sélection de rang, ainsi que l'évaluation de l'atténuation de bruit des filters médians pondérés.

Keywords: Weighted median filter; Optimal filtering; Image processing; Statistical properties

1. Introduction

Weighted median (WM) filters, which were introduced as the generalization of median filters [8], form a subclass of stack filters [16, 20]. Due to their simple operations, their effective suppression of impulsive noise and excellent image detail-preserving capability, WM filters have been successfully applied in various areas from image restoration to image DPCM coding [5, 7, 9, 11, 12, 16, 17, 19].

Since WM filters are nonlinear operations, they do not obey the superposition property and there seems to be no way to give a simple characterization of their behavior. Research has been undertaken from two aspects, namely deterministic properties [4, 6, 15, 18, 21] and statistical properties [1, 17, 20], to study the behavior of WM filters. The basic descriptor of the deterministic properties is a root signal set, a set of signals which pass the filter unaltered. The root signal set resembles the passband of a linear filter. On the other hand, the basic statistical descriptor is the output distribution and the output moments of WM filters. Since WM filters are noise removal operations, noise attenuation capability of WM filters is of great interest. The first attempt to quantize the noise reduction of WM filters was made by Yli-Harja et al. [20] but fell short of being intuitive and efficient. Their result was based on the positive Boolean function representation of WM filters in the binary domain. As a result, all connections to the median operation and the weights of the filter disappear.

Recently, a set of parameters, called M_i, was introduced to characterize WM filters [17]. Expressions for the output distributions and the output moments of WM filters were derived in terms of the M_i's, where the output distributions/moments are expressed as a linear combination of the M_i's. Using the expressions, one can evaluate the noise attenuation of WM filters very efficiently. Most importantly, based on the M_i's, a theory was developed to design optimal WM filters, where the M_i's play an important role in the design process.

Because of the M_i's importance in designing and characterizing WM filters, the computations of the M_i's are of great interest. Using the positive Boolean function representations of WM filters [16, 20], the complexity to compute the M_i's of a WM filter will be of $O(2^N)$, where N is the window size of the WM filter. Astola and Neuvo proposed a method of $O(N^2S)$ to compute the M_i's for integer weights, where S is the sum of the weights [1]. The method works well for integer weights but fails for weights with real number. Unfortunately, in many applications of interesting, weights are of real value.

In this paper, we shall develop two algorithms to compute the M_i's of WM filters. The first one applies for integer weights and the number of operations of the algorithm is about one-fifth of the number of operations of the existing algorithm. The second algorithm works for real weights and reduces the complexity by a great extent for many applications where symmetric weight structures are assumed. In the following section, some basic definitions about WM filters and the M_i's are reviewed, some theorems are presented to show the importance of the M_i's in characterizing statistical properties of WM filters. The new algorithms are presented in Section 3. Applications of the algorithms are discussed in Section 4.

2. Weighted median filters

For the standard median filter, the output is the median of all the samples inside the filter window, thus each sample has the same influence on the output. To achieve new filter properties, one may want to give more weight to samples in specific filter window positions, e.g., to the centermost samples. Weighted median filters were introduced for this purpose [8].

The weighted median filter can be defined in two different but equivalent ways. The most commonly used one assumes positive integer weights with odd sum [8].

Definition 1. For the discrete-time continuous-valued input vector $\boldsymbol{X}$,

$$\boldsymbol{X} = (X_1, X_2, \dots, X_N),$$

the output Y of the WM filter of span N associated with the integer weights

$$\boldsymbol{W} = (W_1, W_2, \dots, W_N),$$

is given by

$$Y = \mathrm{MED}\{W_1 \diamond X_1, W_2 \diamond X_2, \dots, W_N \diamond X_N\}, \tag{1}$$

where $\mathrm{MED}\{\cdot\}$ denotes the median operation and $\diamond$ denotes duplication

$$n \diamond X = \underbrace{X, \dots, X}_{n \text{ times}}.$$

This filtering procedure can be stated as follows: sort the samples inside the filter window, duplicate each sample to the number of the corresponding weight W_i and choose the median value from the expanded sequence.

The second definition of the weighted median allows positive noninteger weights to be used [3].

Definition 2. The weighted median with positive set of weights W of a discrete-time continuous-valued input $\boldsymbol{X}$ is the value β minimizing the expression

$$\Phi(\beta) = \sum_{i=1}^{N} W_i |X_i - \beta|. \tag{2}$$

Here, β is guaranteed to be one of the samples X_i because $\Phi(\cdot)$ is piecewise linear and convex if $W_i \geqslant 0$ for all i.

The output of the WM filter with real positive weights can be calculated as follows: sort the samples inside the filter window (in ascending order), add up the corresponding weights starting from the upper end of the sorted list until the partial sum $\geqslant \frac{1}{2}\sum_{i=1}^{N} W_i$, the output of the WM filter is the sample corresponding to the last weight.

Recently, a set of parameter, called M_i's, was introduced to analyze and design WM filters [17]. It has been shown that statistical properties of WM filters can be expressed in terms of the M_i's and the M_i's are key parameters in design optimal WM filters with structural constraints [17]. In the following, we shall first give the definition of the M_i's.

Definition 3. Consider a WM filter with weight vector $\boldsymbol{W} = (W_1, W_2, \dots, W_N)$. Denote by $\mathscr{W}$ the multiset of weights, i.e.

$$\mathscr{W} = \{W_1, W_2, \dots, W_N\}.$$

Denote by $|\cdot|$ the cardinality operation and by $\Upsilon^{[i]}$ the set of all submultisets of $\mathscr{W}$ having cardinality i, i.e.

$$\Upsilon^{[i]} = \{A \mid A \subseteq \mathscr{W}, |A| = i\}, \quad i = 0, 1, \ldots, N. \tag{3}$$

Let $\Omega^{[i]}$ denote the set of those subsets of cardinality i whose sum of elements is at least the threshold T, i.e.

$$\Omega^{[i]} = \left\{A \mid A \in \Upsilon^{[i]}, \sum_{W_j \in A} W_j \geqslant T\right\}, \quad i = 0, 1, \ldots, N. \tag{4}$$

Such subsets are called *positive subsets* and the threshold T of a WM filter is the half of the sum of the weights, i.e.

$$T = \frac{1}{2} \sum_{i=1}^{N} W_i. \tag{5}$$

In the cases of integer weights, we use the following definition to make the threshold T be an integer,

$$T = \frac{1}{2}\left(1 + \sum_{i=1}^{N} W_i\right). \tag{6}$$

Note that both definitions in (5) and (6) result in same WM filter due to the odd sum of the weights.

In order to illustrate the definition of the positive subsets, consider the following example.

Example 1. Given a WM filter with $\boldsymbol{W} = (1, 4, \underline{5}, 3, 2)$, its threshold is $T = 8$. When $i = 1$, it is obvious that there is no such set which satisfies (4). When $i = 2$, there are two *positive subsets*, i.e.

$$\Omega^{[2]} = \{\{3, 5\}, \{4, 5\}\}.$$

Similarly, one can find other positive subsets:

$$\Omega^{[3]} = \{\{1, 3, 4\}, \{2, 3, 4\}, \{1, 2, 5\}, \{1, 3, 5\}, \{2, 3, 5\}, \{1, 4, 5\}, \{2, 4, 5\}, \{3, 4, 5\}\},$$

$$\Omega^{[4]} = \{\{1, 2, 3, 4\}, \{1, 2, 3, 5\}, \{1, 2, 4, 5\}, \{1, 3, 4, 5\}, \{2, 3, 4, 5\}\},$$

$$\Omega^{[5]} = \{\{1, 2, 3, 4, 5\}\}.$$

Definition 4. Denote by M_i the cardinality of $\Omega^{[i]}$, i.e.

$$M_i = |\Omega^{[i]}|, \quad i = 0, 1, \ldots, N. \tag{7}$$

Example 2. Consider the same WM filter in Example 1, we have

$$M_1 = 0, \qquad M_2 = 2, \qquad M_3 = 8, \qquad M_4 = 5, \qquad M_5 = 1.$$

Three properties of the M_i's are next listed [17]. The first establishes an interesting and useful inter-relation between the M_i's.

Property 1. For a WM filter with window width $N = 2K + 1$, we have

$$M_i + M_{N-i} = \binom{N}{i}, \quad i = 0, 1, \ldots, N. \tag{8}$$

The second property is more intuitive and essential stating the monotonicity of the M_i's.

Property 2. The sequence $M_1, M_2, \ldots, M_K$ is nondecreasing with respect to i, i.e.

$$M_{i+1} \geqslant M_i, \quad i = 1, \ldots, K. \tag{9}$$

This monotonicity property of the M_i's should be intuitive since by increasing the number of weights, more subsets are likely to become positive.

The third property of the M_i's establishes a direct link to the positive Boolean function (PBF) corresponding to the given weighted median. It simply states that M_i is just the cardinality of the subset of the on-set [21], of the PBF representing the WM, containing all true vectors with Hamming weight i.

Property 3. Suppose the binary inputs of a WM filter are vectors

$$\boldsymbol{X} = (X_1, \ldots, X_N) \in \{0,1\}^N.$$

Denote by $f(\boldsymbol{X})$ the positive Boolean function corresponding to this WM. Then, the M_i's can be related to $f(\boldsymbol{X})$ as follows:

$$M_i = |\{\boldsymbol{X} \in \{0,1\}^N \mid f(\boldsymbol{X}) = 1,\ \omega(\boldsymbol{X}) = i\}|, \tag{10}$$

where $\omega(\boldsymbol{X})$ denotes the Hamming weight of $\boldsymbol{X}$ (i.e. the number of 1's in $\boldsymbol{X}$). Eq. (10) simply states that M_i is the cardinality of the subset of the on-set [21], of the PBF representing the WM, containing all true vectors with Hamming weight i.

According to the definition of M_i, one can verify that M_i can be also expressed as follows:

$$M_i = \sum_{\boldsymbol{X} \in S_i} U(\boldsymbol{W}\boldsymbol{X}^{\mathrm{T}} - T), \tag{11}$$

where $U(\cdot)$ is the unit step function, and S_i is a set of all N-dimensional binary vectors with Hamming weight i, i.e.

$$S_i = \{\boldsymbol{X} \mid \boldsymbol{X} \in \{0,1\}^N;\ \omega(\boldsymbol{X}) = i\}. \tag{12}$$

The following theorems show that the output distributions and output moments of WM filters, when applied to i.i.d. inputs, can be expressed as the linear combinations of the M_i's [17].

Theorem 1. *Let the inputs of a WM filter with window width $N = 2K + 1$ be i.i.d. with a common distribution function $\Phi(t)$. The output distribution of the WM filter $\Psi_{\mathrm{wm}}(t)$ has the following form:*

$$\Psi_{\mathrm{wm}}(t) = \Psi_{\mathrm{s}}(t) + \sum_{i=1}^{K} M_i(\Phi(t)^i(1 - \Phi(t))^{N-i} - \Phi(t)^{N-i}(1 - \Phi(t))^i), \tag{13}$$

where $\Psi_{\mathrm{s}}(t)$ is the output distribution of the standard median filter with the same window width, i.e.

$$\Psi_{\mathrm{s}}(t) = \sum_{i=K+1}^{N} \binom{N}{i} \Phi(t)^i(1 - \Phi(t))^{N-i}.$$

Theorem 2. *Given a WM filter with window width $N = 2K + 1$, for i.i.d. inputs with common distribution function $\Phi(t)$ and density function $\phi(t)$, the γ-order output central moment, denoted by $\mu_{\mathrm{wm}}^{\gamma}$, of the WM filter can be expressed as*

$$\mu_{\mathrm{wm}}^{\gamma} = \mu_{\mathrm{s}}^{\gamma} + \sum_{i=1}^{K} M_i L_i(N, \Phi, \gamma), \tag{14}$$

where μ_s^γ is the γ-order central moment of the standard median with the same window size,

$$L_i(N,\Phi,\gamma)=\int_{-\infty}^{+\infty}U_i(\Phi(y))|y-m_y|^\gamma\phi(y)\,\mathrm{d}y\geqslant 0 \tag{15}$$

for symmetric ϕ, $\gamma\geqslant 0$, $i=1,\dots,K$, m_y is the output mean and

$$U_i(\Phi)=(i-N\Phi)\Phi^{i-1}(1-\Phi)^{N-i-1}+(i-N(1-\Phi))\Phi^{N-i-1}(1-\Phi)^{i-1}.$$

Moreover, the M_i's have been proved as the critical parameters to design optimal WM filters with structural approach and to calculate the rank selection probabilities [10, 17].

3. Fast algorithms

Since the M_i's play a critical role in designing and analyzing WM filters, the computation of the M_i's becomes an important issue. In the following two subsections, we shall propose two fast algorithms to compute the M_i's of WM filters, one for integer weights, another for real weights.

3.1. Algorithm 1

In principle, M_i's can be computed using the positive Boolean function representations of WM filters. The complexity of the procedure is of $O(2^N)$. Recently, Astola and Neuvo proposed a fast algorithm, whose complexity is of $O(N^2S)$, to compute the M_i's of WM filters with integer weights, where S is the sum of the weights.

In the following, we will propose a quite efficient method, whose number of operations is less than one-fifth of that of the algorithm proposed in [1].

Suppose a WM filter with integer weights $\boldsymbol{W}=(W_1,W_2,\dots,W_N)$. Consider the following polynomial:

$$F(Y,Z)=\prod_{j=1}^{N}(1+YZ^{W_j}). \tag{16}$$

When expanded it contains 2^N terms, each of the form Y^iZ^l. These can be thought of as combinations of i weights whose sum is l. Therefore, the 2^N terms of the polynomial in Eq. (16) represent all weight combinations with 1 to N weights. Now, expand the polynomial in Eq. (16) in powers of Y^iZ^l as follows:

$$F(Y,Z)=\sum_{i=0}^{N}\sum_{l=1}^{S}F_{i,l}Y^iZ^l, \tag{17}$$

where S is the sum of the weights

$$S=\sum_{i=1}^{N}W_i.$$

It can be shown that M_i can be expressed as the sum of the coefficients $F_{i,l}$ for $l\geqslant T$, i.e.

$$M_i=\sum_{l=T}^{S}F_{i,l}. \tag{18}$$

Now the problem to compute the M_i's becomes the problem to compute these $F_{i,l}$'s. Obviously, the polynomial $F(Y,Z)$ can be written in a recursive form,

$$F^{[i]}(Y,Z) = (1 + YZ^{W_i})F^{[i-1]}(Y,Z), \quad i = 1, \dots, N, \tag{19}$$

with the initial value $F^{[0]}(Y,Z) = 1$. The algorithm to compute the M_i's is shown as follows.

Algorithm 1

(i) Sort $(W_1, W_2, \dots, W_N)$ in ascendent order.
(ii) $f^{[0]}(1,1) = 1$
(iii) Compute $f(j,l)$

```
for i = 1 to N
  if i < K + 1, then
    for j = 1 to i + 1
      for l = 1, 1 + Σ_{m=1}^{i} W_m
        f^[i](j,l) = f^[i-1](j,l) + f^[i-1](j - 1, l - W_i)
      end
    end
  else
    for j = 1 to K + 1
      for l = 1, 1 + Σ_{m=i-K+1}^{i} W_m
        f^[i](j,l) = f^[i-1](j,l) + f^[i-1](j - 1, l - W_i)
      end
    end
end
```

(iv) Compute M_i's

```
for i = 1 to (N - 1)/2
  M(i) = 0
  for j = T to S
    M(i) = M(i) + f(i + 1, j)
  end
end
```

Note that the first sorting step is not trivial. It will reduce the number of operations unless all weights are equal to each other.

Theorem 3. *The number of the operations of above algorithm is less than* $(\frac{3}{16}N^2S)$.

The proof is in Appendix A.

Note that the number of operations of our algorithm is less than one-fifth of that of Astola and Neuvo's algorithm, whose number of operations is N^2S. The major difference between these two algorithms is the way to calculate the matrix $f(i,j)$. In our algorithm, the inter-relationship between M_i's is utilized, i.e. only the first $K + 1$ columns of $f(i,j)$ matrix need to be calculated. Secondly, only nonzero elements in the matrix are updated. That saves a number of operations.

Example 3. In order to demonstrate how the algorithm works, consider a 5-point WM filter with weight vector $W = (1, 2, \underline{3}, 2, 1)$. In the following we list all $f(j,l)$ matrices in each iteration (for the purpose of illustration the full $f(i,j)$ matrices are shown here, even though only their fast $K + 1$ columns need to be calculated in the algorithm). The elements in the first row and the first column locate at low left corner of the matrices.

Note that weights are first sorted in ascendent order, i.e. $\boldsymbol{W} = (1, 1, 2, 2, 3)$.

						0 0 0 0 0 1	
						0 0 0 0 2 0	
						0 0 0 1 2 0	
					0 0 0 0 1	0 0 0 4 1 0	
Matrices					0 0 0 2 0	0 0 2 3 0 0	$T = 5$
				0 0 0 1	0 0 1 2 0	0 0 3 2 0 0	
				0 0 2 0	0 0 4 0 0	0 1 4 0 0 0	
			0 0 1	0 1 1 0	0 2 1 0 0	0 2 1 0 0 0	
		0 1	0 2 0	0 2 0 0	0 2 0 0 0	0 2 0 0 0 0	
	1	1 0	1 0 0	1 0 0 0	1 0 0 0 0	1 0 0 0 0 0	
Column no.	1	1 2	1 2 3	1 2 3 4	1 2 3 4 5	1 2 3 4 5 6	
Iteration	0	1	2	3	4	5	
New weight		$W_1 = 1$	$W_2 = 1$	$W_3 = 2$	$W_4 = 2$	$W_5 = 3$	
M's						$M_1 = 0$	
						$M_2 = 2$	
						$M_3 = 8$	
						$M_4 = 5$	
						$M_5 = 1$	

The M_i's are computed from the final $f(j, l)$ matrix in the last iteration. Note that M_i is calculated from column $i + 1$ of $f(j, l)$ matrix. For instance, in order to calculate the M_3 of the example, simply add all elements in column 4 which are above the threshold T, here these elements are 3, 4, 1 and two 0's, thus $M_3 = 8$.

Using the binomial theorem, we can prove the following theorem.

Theorem 4. *The sum of all elements in the final $f(j, l)$ matrix is 2^N, where N is the window size, for any integer valued weight vector.*

Note that the efficiency of the algorithm relies on the fact that the sum of the given weights is minimized. In that case, these nonzero elements in $f(j, l)$ matrix are overlapped, making the dimensions of the matrix be compressed and reducing the number of operations. When the sum of the given weight vector is not minimized, the algorithm may not be efficient, sometime may have high computation complexity. For example, consider a 5-point WM filter with weight $\boldsymbol{W} = (1, 2, 4, 8, 16)$. One can verify that there is no overlapping among these nonzero elements in the $f(j, l)$ matrix. The computation complexity becomes $O(N^2 2^N)$ in the case. Obviously, this WM filter is an identity filter with minimum-of-sum representation $\boldsymbol{W} = (1, 1, 5, 1, 1)$. Therefore, Algorithm 1 is not suitable when the sum of the given weights is not minimized. Moreover, Algorithm 1 can not be applied to the WM filters whose weights are of real value.

Note that the Algorithm 1 works also for rank order filters, including weighted order statistic filters, by changing the threshold T to the appropriate value.

3.2. Algorithm 2

As we mentioned earlier, Algorithm 1 is restricted by the fact that it cannot be used for computation of M_i's for real weights. Unfortunately, in many applications of interest, the weights of WM filters are of real number. For instance, in adaptive WM filtering [19] and optimal WM filtering [17], where weights are updated in each iteration and are no longer integers. Therefore, it is of importance to study algorithms for computing M_i's of real weights.

There are, usually, two ways to do so. One way is to convert real weights into integer weight representation before using Algorithm 1, by resorting to the positive Boolean function (PBF) representation and integer programming. It can be very much computationally complex. Research has been undertaken in this direction, but so far not much progress has been made [13]. Another way is to check the PBF representation of WM filters for 2^N entries. This method is quite intuitive and simple when N, the window size of WM filters, is small. It has 2^N iterations and in each iteration one binary vector is checked. The problem arises from these binary vectors. When N is large, for instance 25, we cannot store these 2^{25} ($32M$) vectors in our computer. Instead, we have to generate these vectors in each iteration. It definitely will increase the number of operations dramatically.

On the other hand, in many applications, symmetric weight structures of WM filters are often used, making the number of parameters reduced. However, the method which use the PBF representation of WM filters cannot take the advantage. In the following, using the similar principle of Algorithm 1, we develop a fast algorithm for computing the M_i's of WM filters having real weights.

Algorithm 2

(i) Sort $(W_1, W_2, \dots, W_N)$ in ascendent order.
(ii) $f^{[0]}(1,1) = 1$, $L(1) = 0$, Level = 1
(iii) Compute $f(i,j)$

```
  for i = 1 to N
    for j = 1 to level
      for l = 1, to i
        f^[i](j,l) = f^[i-1](j,l)
      end
      f^[i](j,i + 1) = 0
    end
    for j = level + 1 to 2*level
      f^[i](j,1) = 0
      for l = 2, to i + 1
        f^[i](j,l) = f^[i-1](j - level,l - 1)
      end
    end
    for j = level + 1 to 2*level
      L(j) = L(j - level) + W_i
    end
    sort sequence in L
    compress L
    compress f(j,l)
    level = number of rows in f(j,l)
  end
```

(iv) Compute M_i's

```
  for i = 1 to (N - 1)/2
    M(i) = 0
    for j = T to S
      M(i) = M(i) + f(i + 1,j)
    end
  end
```

In the algorithm, *compress* L and *compress* $f(j,l)$ mean to merge these elements which have identical value in L and then merge the corresponding rows in $f(j,l)$ matrix.

Example 4. Consider a WM filter with $W = (0.1, 0.3, 0.5, 0.3, 0.1)$. The weights are first sorted in ascendent order $(0.1, 0.1, 0.3, 0.3, 0.5)$. The $f(j,l)$ matrices in each iteration are listed as follows.

							$L(j)$
						0 0 0 0 0 1	1.3
						0 0 0 0 2 0	1.2
						0 0 0 1 0 0	1.1
						0 0 0 0 2 0	1.0
						0 0 0 4 0 0	0.9
					0 0 0 0 1	0 0 2 0 1 0	0.8
Matrices					0 0 0 2 0	0 0 0 3 0 0	$T = 0.7$
					0 0 1 0 0	0 0 3 0 0 0	0.6
				0 0 0 1	0 0 0 2 0	0 1 0 2 0 0	0.5
				0 0 2 0	0 0 4 0 0	0 0 4 0 0 0	0.4
				0 1 0 0	0 2 0 0 0	0 2 0 0 0 0	0.3
			0 0 1	0 0 1 0	0 0 1 0 0	0 0 1 0 0 0	0.2
		0 1	0 2 0	0 2 0 0	0 2 0 0 0	0 2 0 0 0 0	0.1
	1	1 0	1 0 0	1 0 0 0	1 0 0 0 0	1 0 0 0 0 0	0
Column no.	1	1 2	1 2 3	1 2 3 4	1 2 3 4 5	1 2 3 4 5 6	
Iteration	0	1	2	3	4	5	
New weight		$W_1 = 0.1$	$W_2 = 0.1$	$W_3 = 0.3$	$W_4 = 0.3$	$W_5 = 0.5$	
M's						$M_1 = 0$	
						$M_2 = 2$	
						$M_3 = 8$	
						$M_4 = 5$	
						$M_5 = 1$	

In order to demonstrate the algorithm, we work through the iteration No. 4. At the beginning of iteration 4, we got $f(j,l)$ matrix and $L(j)$ sequence from iteration 3, which are listed as follows, level = 6:

$f^{[3]}(j,l)$	$L^{[3]}(j)$
0 0 0 1	0.5
0 0 2 0	0.4
0 1 0 0	0.3
0 0 1 0	0.2
0 2 0 0	0.1
1 0 0 0	0

The coming weight in iteration 4 is $W_4 = 0.3$. First we build a 5×12 matrix (i.e. $(i+1) \times (2 \times \text{level})$) by placing an $f^{[3]}(j,l)$ matrix on the top of another $f^{[3]}(j,l)$ matrix and shift one column to the right, and other empty places are filled by 0's. The $L(j)$ vector is constructed by placing a vector, whose elements $L^{(3)}(j)$'s elements plus $W_4 = 0.3$, on the top of another $L^{[3]}(j)$, i.e.

$f(j,l)$		$L(j)$
0	0 0 0 1	$0.5 + 0.3 = 0.8$
0	0 0 2 0	$0.4 + 0.3 = 0.7$
0	0 1 0 0	$0.3 + 0.3 = 0.6$
0	0 0 1 0	$0.2 + 0.3 = 0.5$
0	0 2 0 0	$0.1 + 0.3 = 0.4$
0	1 0 0 0	$0 + 0.3 = 0.3$
0 0 0 1	0	0.5
0 0 2 0	0	0.4
0 1 0 0	0	0.3
0 0 1 0	0	0.2
0 2 0 0	0	0.1
1 0 0 0	0	0

Then $L(j)$ is sorted and elements which have identical value are merged into one. And these rows in $f(j,l)$ which correspond to identical value of $L(j)$ are added together and become one new row. The new $f(j,l)$ matrix, denoted by $f^{[4]}(j,l)$, is obtained by arranging these new rows to make their associated elements in $L(j)$ vector in an ascendent order, i.e.

row no.	$f^{[4]}(j,l)$	$L^{[4]}(j)$
9	0 0 0 0 1	0.8
8	0 0 0 2 0	0.7
7	0 0 1 0 0	0.6
6	0 0 0 2 0	0.5
5	0 0 4 0 0	0.4
4	0 2 0 0 0	0.3
3	0 0 1 0 0	0.2
2	0 2 0 0 0	0.1
1	1 0 0 0 0	0

Note also that only the first $K + 1$ columns of $f(j,l)$ matrix are needed for calculating the M_i's.

It can be shown that the complexity of Algorithm 2 is of $O(N^2 2^L)$, where L is the number of distinct weights in the filter's window. Usually, L is much smaller than N in practice. This makes the algorithm very efficient in these applications where symmetric weight structures are used. For example, to compute the M_i's of a 5×5 WM filter having symmetric weights with respect to horizontal and vertical axes, there are only five distinct weights while the number of weights is 25. Thus the fast algorithm is about one thousandth of the existing algorithm in terms of computational complexity. Note that the storage requirement depends on 2^L instead of 2^N.

4. Applications

It has been shown that the output distribution and output moments of WM filters are linear combinations of the M_i's. Therefore, the statistical properties of WM filters can be analyzed using the M_i's. Since WM filters usually are noise-removal filters, the evaluation of noise reduction of WM filters is of great interest.

Recently, a theory was developed for optimal WM filtering with structural constraints [17]. The problem arises when a set of pre-specified constraints is imposed on filter's behavior, a WM filter is sought to yield maximum noise attenuation. For instance, an image consists of many signal structures, such as lines and corners, which are critical to perception. When filtering noisy images, care must be taken not to remove such important image details. WM filters have such an attractive feature that they can be designed to preserve the desired image details by selecting an appropriate set of weights. However, for a given set of image details, there may be a number of WM filters. The problem is how to choose one of them which has maximum noise attenuation. The problem is referred as *Optimal WM Filtering with Structural Constraints* [2, 17]. In the newly developed theory, it has been shown that optimal WM filters can be designed by resorting to nonlinear programming. Again, the parameters M_i's become the key to solve the problem.

In order to make the paper self-contained, here we briefly review some results in the theory of optimal WM filtering [17].

Denote by $X(i,j)$ the input sample of a constant signal s corrupted by some additive white noise n to be filtered by a WM with weight vector $\boldsymbol{W}$. Then the output $\hat{s}$ of the WM filter, which is an estimate of s, can be written as

$$\hat{s}(i,j) = \mathrm{MED}\{W_{s,t} \diamond X(i-s, j-t) \,|\, (s,t) \in W\}. \tag{20}$$

Take the mean square error (MSE) as the criterion function

$$J = E(s - \hat{s})^2. \tag{21}$$

According to statistical properties of WM filters [17], we have the following theorem.

Theorem 5. *For WM filters with window size $N = 2K + 1$, the mean square error J defined in (21) can be expressed as*

$$J = \sigma_s^2 + \sum_{i=1}^{K} L_i M_i, \tag{22}$$

where σ_s^2 and L_i*'s are independent of the weights and*

$$L_i \geqslant 0, \quad i = 1, \ldots, K.$$

From this theorem it is concluded that designing optimal WM filters under structural constraints is equivalent to minimizing the M_i's under detail-preserving requirements, i.e.

$$\text{Minimize} \quad \sum_{i=1}^{K} L_i M_i \quad \text{subject to structural constraints.} \tag{23}$$

By Eq. (11) the M_i's can be expressed as functions of the weights in the form of the unit-step function. Then we approximate the unit-step function by a sigmoidal function

$$U_s(z) = \frac{1}{1 + \mathrm{e}^{-\beta z}}. \tag{24}$$

$U_s(\cdot)$ is a continuous differentiable, monotonically increasing, step-like function. Its steepness is controlled by a gain term $\beta > 0$. When gain β is large, the unit-step function can be approximated well by the sigmoidal function. Thus the optimal problem can be restated as

$$\begin{aligned} &\text{Minimize} \quad \sum_{i=1}^{K} L_i M_i = \sum_{i=1}^{K} \sum_{x \in S_i} L_i U_s(\boldsymbol{W}\boldsymbol{x}^{\mathrm{T}} - T) \\ &\text{subject to} \quad \boldsymbol{W} \geqslant 0 \ \text{(positivity constraints)}, \\ &\qquad\qquad\quad \boldsymbol{C}\boldsymbol{W} \geqslant 0 \ \text{(structural constraints)}. \end{aligned} \tag{25}$$

where S_i is defined in Eq. (12) and matrix $\boldsymbol{C}$ represents the structural constraints. This is a nonlinear programming problem in which the objective function has first and second derivatives. It can be solved by successive quadratic programming method [14].

Note that in solving the nonlinear programming problem, the weight vector $\boldsymbol{W}$ has to be updated in each iteration. Both the objective function and its first derivative have to be computed in each iteration and thus 2^N sigmoidal functions need to be calculated, making the problem very high computational complexity. Instead, if we only use sigmoidal functions in deriving the first derivative of the objective function but use their M_i's representation in computing both the objective function and its first derivative, the problem becomes much simpler. In this case, M_i's have to be computed for real weights. When N becomes larger, the computational complexity of the conventional method to compute M_i's, i.e. positive Boolean function representations of WM filters, will be very high. The fast algorithm developed in this paper, Algorithm 2, can be applied here to greatly shorten the design time. See the following example.

Example 5. Design a 5×5 optimal WM filter which preserves horizontal lines, vertical lines and diagonal lines. From practical point of view, symmetric weight structure is assumed.

Consider 5×5 WM filters of the following form:

$$\boldsymbol{W} = \begin{pmatrix} W_1 & W_2 & W_3 & W_4 & W_5 \\ W_6 & W_7 & W_8 & W_9 & W_{10} \\ W_{11} & W_{12} & W_{13} & W_{14} & W_{15} \\ W_{16} & W_{17} & W_{18} & W_{19} & W_{20} \\ W_{21} & W_{22} & W_{23} & W_{24} & W_{25} \end{pmatrix}.$$

The structural constraints can be written as follows:

$$\begin{aligned} &W_{11} + W_{12} + W_{13} + W_{14} + W_{15} \geqslant T, \\ &W_3 + W_8 + W_{13} + W_{18} + W_{23} \geqslant T, \\ &W_1 + W_7 + W_{13} + W_{19} + W_{25} \geqslant T, \\ &W_5 + W_9 + W_{13} + W_{17} + W_{21} \geqslant T. \end{aligned} \tag{26}$$

Note that the four inequalities in Eq. (26) correspond to horizontal lines, vertical lines and two diagonal lines, respectively.

The following symmetric weight structure is used:

$$\begin{aligned} &W_1 = W_5 = W_{21} = W_{25}, \\ &W_2 = W_4 = W_6 = W_{10} = W_{16} = W_{20} = W_{22} = W_{24}, \\ &W_3 = W_{11} = W_{15} = W_{23}, \\ &W_7 = W_9 = W_{17} = W_{19}, \\ &W_8 = W_{12} = W_{14} = W_{18}. \end{aligned} \tag{27}$$

Together with the positivity constraints of the weights

$$W_i \geqslant 0 \tag{28}$$

for $i = 1, \ldots, 25$.

The optimal problem becomes

$$\text{Minimize} \quad \sum_{i=1}^{12} L_i M_i = \sum_{i=1}^{12} \sum_{x \in S_i} L_i U_s(\boldsymbol{W}\boldsymbol{x}^{\mathrm{T}} - T) \quad \text{subject to (26)–(28)}. \tag{29}$$

We utilize *International Mathematics and Statistics Library* to solve this nonlinear programming problem [14]. As stated earlier, we need to supply several subroutines to compute the objective function and its derivative for a given weight vector in each iteration. In this process, M_i's need to be calculated in order to calculate the objective function and its derivative. Since the given weight vector is of real value, we have to use positive Boolean representation of WM filters to check 2^N entries to calculate the M_i's before Algorithm 2 is developed. It takes tens of hours to get the final results. We now apply Algorithm 2 to calculate the M_i's. It takes less than one minute to get the final result! It sounds reasonable. Because of the symmetric structure of weights, there are only five variables. $O(N^2 \times 2^5)$ of Algorithm 2 compared against to $O(2^N)$ of the conventional method, the reduction in complexity is obvious.

The program is written in Fortran Language and run at MIPS RC6280.

The final solution is

$$\boldsymbol{W} = \begin{pmatrix} 0.249 & 0 & 0.169 & 0 & 0.249 \\ 0 & 0.249 & 0.333 & 0.249 & 0 \\ 0.169 & 0.333 & 2.08 & 0.333 & 0.169 \\ 0 & 0.249 & 0.333 & 0.249 & 0 \\ 0.249 & 0 & 0.169 & 0 & 0.249 \end{pmatrix}.$$

In order to demonstrate the performance of the optimal WM filter obtained in the example, we apply the WM filter to filter a noisy image corrupted by impulsive noise. The probability of impulses was 0.04 and the height of the impulses was set to ± 200. Using the original and its corrupted version, the 5×5 Wiener filter was obtained and applied to restore the noisy image. The MAE and MSE values are listed in Table 1. These filters are also applied to restore image "Bridge" and "Harbor" which were corrupted by the same noise. Results are included in Table 1. Fig. 1 shows the original, noisy and filtered images of part of Lenna's right eye.

The fast algorithms can also be used for other purposes, for instance, to calculate the rank selection probabilities of WM filters. It can reduce the complexity from previous $O(N!)$ to $O(N^2 S)$ or $O(N^2 2^L)$.

Table 1
Filters' performance in impulsive noise

	Bridge		Lenna		Harbor	
Filter	MSE	MAE	MSE	MAE	MSE	MAE
Wiener filter	350.7	14.37	223.1	11.60	302.5	12.54
Median filter	261.2	10.38	73.5	4.79	294.7	8.73
WM filter	85.2	3.34	30.4	1.63	87.1	2.85

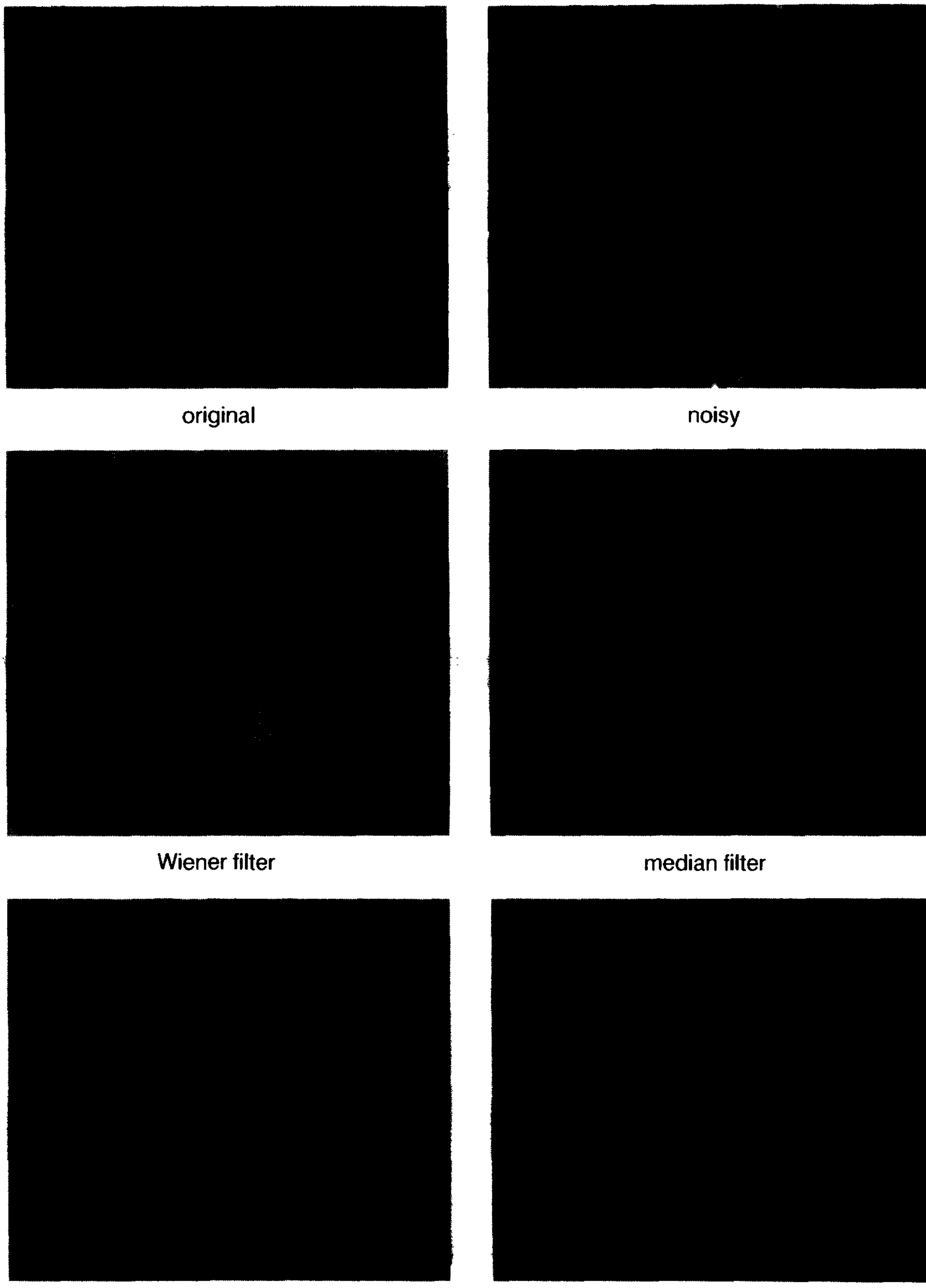

Fig. 1. The original, noisy and filtered images of Lenna's right eye.

5. Conclusions

In the paper we have developed two fast algorithms to calculate M_i's, the parameters that play a very important role in analyzing and designing WM filters. The first algorithm works for integer weights and is about four times fast over the existing algorithm. The second fast algorithm is proposed to calculate the M_i's of weights with real value. This is critical for applications since in practice many weight vectors of interest are of real value. Some examples were presented to demonstrate the efficiency of the algorithms.

Appendix A

Proof of Theorem 3. Denote the number of operations of the proposed algorithm by $C(N)$. And it can be expressed as

$$C(N) = \sum_{i=1}^{K} i \sum_{j=1}^{i} W_j + \sum_{i=K+1}^{N} K \sum_{j=i-K+1}^{i} W_j .$$

There are N iterations in the algorithm. The first term comes from the first K iterations, while the second comes from the rest of iterations. $C(N)$ can be rewritten as

$$C(N) = K \sum_{i=K+1}^{N} \sum_{j=1}^{i} W_j + \left(\sum_{i=1}^{K} \sum_{j=1}^{i} (i-K) W_j - K \sum_{j=1}^{K+1} W_j \right).$$

Obviously, the term in parenthesis is less than zero, which means that $C(N)$ is less than the first term, denoted by $C1(N)$, i.e.

$$C(N) < C1(N) = \sum_{i=K+1}^{N} \sum_{j=1}^{i} W_j .$$

Our goal here, is to find an upper bound of $C(N)$. We can take the maximum of $C1(N)$ as an upper bound of $C(N)$. $C1(N)$ can be written as

$$C1(N) = \sum_{i=1}^{N} C_i W_i ,$$

where

$$C_i = \begin{cases} K(K+1) & \text{for } 1 \leqslant i \leqslant K+1, \\ K(N-i+1) & \text{for } K+2 \leqslant i \leqslant N. \end{cases}$$

Since

$$C_i \geqslant C_j \quad \text{for } i \leqslant j ,$$

and remember the condition

$$W_i \leqslant W_j \quad \text{for } i \leqslant j ,$$

we come to the conclusion that $C1(N)$ will reach its maximum when all weights are equal, i.e.

$$W_1 = W_2 = \cdots = W_N = \frac{S}{N} .$$

The maximum is given by

$$\sum_{i=1}^{K+1} K(K+1)W_i + \sum_{i=K+2}^{N} K(N-i+1)W_i$$

$$= \frac{S}{N}K(K+1)\sum i = 1^{K+1} + \frac{S}{N}\sum_{i=K+2}^{N} K(N+1-i)$$

$$= \frac{S}{N}K(K+1)^2 + \frac{S}{N}K\left(K(N+1) - \frac{1}{2}K(N+K+2)\right)$$

$$= \frac{S}{N}K(K+1)^2 + \frac{1}{2}\frac{S}{N}K^2(K+1)$$

$$\approx \frac{1}{8}N^2S + \frac{1}{16}N^2S$$

$$= \frac{3}{16}N^2S.$$

It follows that the number of operations of Algorithm 1 is less than $\frac{3}{16}N^2S$. □

References

[1] J. Astola and Y. Neuvo, "An efficient tool for analyzing weighted median and stack filters", *IEEE Trans. Circuits and Systems*, to appear.

[2] E.J. Coyle, J-H Lin and M. Gabbouj, "Optimal stack filtering and the estimation and structural approaches to image processing", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. ASSP-37, No. 12, December 1989, pp. 2037–2066.

[3] V.J. Gebski, "Some properties of splicing when applied to nonlinear smoothers", *Comput. Statist. Data Analysis*, No. 3, 1985, pp. 151–157.

[4] P. Haavisto, M. Gabbouj and Y. Neuvo, "Median based idempotent filters", *J. Circuits Systems Comput.*, Vol. 1, No. 2, June 1991, pp. 125–148.

[5] P. Haavisto, J. Juhola and Y. Neuvo, "Motion adaptive scan rate up-conversion", (*Invited paper*) *Multidimensional Systems and Signal Processing J., Special Issue on Multidimensional Processing of Video Signals*, Vol. 3, Nos. 2/3, May 1992, pp. 113–130.

[6] Y. Han, I. Song and Y. Park, "Some root properties of recursive weighted median filters", *Signal Processing*, Vol. 25, No. 3, December 1991, pp. 337–344.

[7] T. Jarske and Y. Neuvo, "Adaptive DPCM with median type predictors", *IEEE Trans. Consumer Electronics*, Vol. 37, No. 3, August 1991, pp. 348–352.

[8] B.J. Justusson, "Median filtering: statistical properties", in: T.S. Huang, ed., *Two Dimensional Digital Signal Processing II*, Springer, Berlin, 1981, pp. 161–196.

[9] S.-J. Ko and Y.H. Lee, "Center weighted median filters and their application to image enhancement", *IEEE Trans. Circuits and Systems*, Vol. 38, No. 9, September 1991, pp. 984–993.

[10] P. Kuosmanen and J. Astola, On rank selection probabilities, Research Report of University of Tampere, A 272, January, 1993.

[11] Y.H. Lee, D.H. Kang, J.H. Choi and K.D. Lee, "DPCM with median predictors", in: E.R. Dougherty, J. Astola and C. Boncelet Jr., eds., *Nonlinear Image Processing III, Proc. SPIE*, Vol. 1658, February 1992, pp. 199–209.

[12] T. Loupas, W.N. McDicken and P.L. Allan, "An adaptive weighted median filter for speckle suppression in medical ultrasonic images", *IEEE Trans. Circuits and Systems*, Vol. CAS-36, January 1989, pp. 129–135.

[13] J. Nieweglowski, M. Gabbouj and Y. Neuvo, "Weighted medians – Positive Boolean functions conversion algorithms", *Signal Processing*, Vol. 34, No. 2, November 1993, pp. 149–161.

[14] User's Manual, Math/Library, Problem-solving software systems, version 1.0, April 1987.

[15] P.D. Wendt, "Nonrecursive and recursive stack filters and their filtering behavior", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. ASSP-38, No. 12, December 1990, pp. 2099–2107.

[16] P.D. Wendt, E.J. Coyle and N.C. Gallagher Jr., "Stack filter", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. ASSP-34, No. 4, August 1986, pp. 898–911.

[17] R. Yang, L. Yin, M. Gabbouj, J. Astola and Y. Neuvo, "Output distributions of weighted median filters and optimization with structural constraints", *IEEE Trans. Signal Process.*, to appear. Part of the results was presented at IEEE Int. Symp. on CAS, 3–6 May 1993, Vol. 1, Chicago, Illinois, USA, pp. 942–945.

[18] R. Yang, M. Gabbouj and Y. Neuvo, "On convergence behavior of symmetric weighted median filters", *Proc. European Conference on Circuit Theory and Design*, Davos, Switzerland, 30 August–3 September 1993, pp. 1053–1058.

[19] L. Yin, J. Astola and Y. Neuvo, "Adaptive stack filtering with application to image processing", *IEEE Trans. Signal Process.*, Vol. 41, No. 1, January 1993, pp. 162–184.

[20] O. Yli-Harja, J. Astola and Y. Neuvo, "Analysis of the properties of median and weighted median filters using threshold logic and stack filter representation", *IEEE Trans Acoust. Speech Signal Process.*, Vol. 39, No. 2, February 1991, pp. 395–410.

[21] Pao-Ta Yu and E. Coyle, "On the existence and design of the best stack filter based associative memory", *IEEE Trans. Circuits and Systems*, Vol. 39, No. 3, March 1992, pp. 171—184.