THE METHODS OF DESIGN AND IMPLEMENTATION OF STACK FILTERS FOR IMAGE PROCESSING

DRUTAROVSKÝ, M. - MARCHEVSKÝ, S.
Department of Radioelectronics
FEI TU in Košice
Park Komenského 13
040 21 Košice
Slovakia

Abstract

This paper deals with a large class of nonlinear digital filters, the stack filters, which contain all combinations and compositions of rank order operators within a finite window. Attention is given to design and effective hardware implementation of an optimal stack filter for image processing. Presented simulation results confirm robustness of stack filters in the image restoration corrupted by impulsive noise.

Keywords:

Nonlinear digital filters, stack filters, image processing, neural networks.

1. Introduction

Rank order filters have been used in both signal and image applications. These filters work well in many situations in which other filtering techniques either fail or are inappropriate. For instance, they can remove impulsive noise such as speckle noise, from images without blurring the sharp edges usually occurred in images [1]. Great advances have been made in the design and implementation of stack filters (SF's), which constitute a broad class of nonlinear filters based on order statistics [2-7].

2. The definition of stack filters

A SF's are nonlinear digital filters with sliding window. Each filter in this class possesses the weak superposition property known as the threshold decomposition (TD) and an ordering property known as the stacking property [2]. For simplicity we will use notation for 1D processes. Extension to 2D processes is straightforward. Let $X(n)$ be an arbitrary time discrete bounded signal. Let $\mathbf{X}_N = [X_1, X_2, ..., X_N]^T = [X(n), X(n-1), ..., X(n-N+1)]^T$ denote sampled and to M levels quantized version of $X(n)$. Then the TD of X_j (M-level value), called threshold signals x_j^m (2-level values) whose elements are defined by

$$x_j^m = T^m(X_j) = \begin{cases} 1 & \text{if } X_j \geq m \\ 0 & \text{if } X_j < m \end{cases} \qquad \begin{matrix} m = 1,2,...,M-1 \\ j = 1,2,...,N \end{matrix} \qquad (1)$$

In Fig. 1 we show the TD architecture of SF.

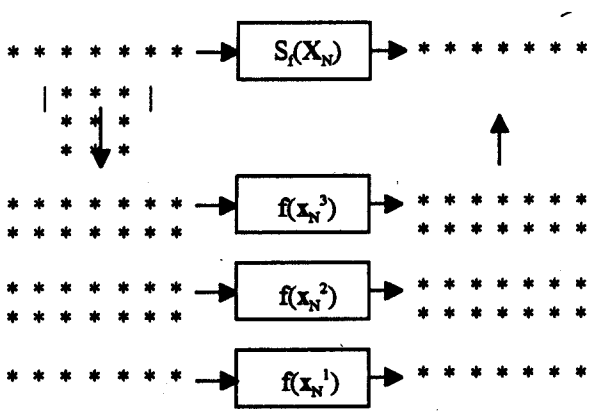

Fig.1
The TD architecture of SF for M=4 and N=3

Mathematically Fig.1 states the following superposition property for the SF $S_f(.)$ based on the binary function $f(.)$:

$$S_f(\mathbf{X}_N) = S_f\left(\sum_{m=1}^{M-1} T^m(\mathbf{X}_N)\right) = \sum_{m=1}^{M-1} S_f(T^m(\mathbf{X}_N)) =$$

$$= \sum_{m-1}^{M-1} \hat{s}^m(n) = \sum_{m-1}^{M-1} f(\mathbf{x}_N) \qquad (2)$$

The TD states that stack filtering of multilevel signal is the same as the first decomposing it into a set of binary signals by thresholding then filtering each binary signal with a binary SF, and, finally adding the results of these operations. The second property of SF's is called the stacking property, which is an ordering property. It is best described as follows: if the binary output signal of each of the Boolean operators on the threshold level in Fig.1 are piled on top of each other according to their threshold level, the result is always a column of 1's supporting a column of 0's. It has been proven that the necessary and sufficient condition for binary function $f(.)$ to possess the stacking property is that it is positive Boolean function

(PBF) [2]. The number of SF's for any fixed window width N is at least 2^y, where $y = 2^{\lceil N/2 \rceil}$ [2,3], where $\lceil x \rceil$ is the smallest integer larger than or equal to x.

3. The design of stack filters

3.1 The optimal stack filtering

The optimal problem over the class of SF's can be stated as shown in Fig.2.

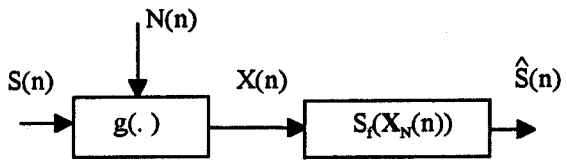

Fig.2
The optimal filtering problem

The process $X(n)$ (e.g. a corrupted image) at the input of SF is assumed to be a corrupted version of desired process $S(n)$ (e.g. an original image). The corruption is a noise process $N(n)$. At each time instant n, the SF output is an estimate called $\hat{S}(n)$ of the desired process $S(n)$. This estimate is based on the observed sequence $\mathbf{X}_N(n)$ in the window of the SF, thus

$$\hat{S}(n) = \sum_{m=1}^{M-1} \hat{s}^m(n) = S_f(\mathbf{X}_N(n)) \qquad (3)$$

The goal of optimal stack filtering is to find a SF from the class of SF's with window width N such that the mean absolute error (MAE) between the filter output and the desired signal is minimized. If $S(n)$ and $X(n)$ are jointly stationary, then the cost to be minimized is

$$E[|S(n) - S_f(\mathbf{X}_N(n))|] = \sum_{m-1}^{M-1} E[|s^m(n) - f(T^m(\mathbf{X}_N(n)))|] \qquad (4)$$

The optimal filtering problem over the class of SF's under the MAE criterion can therefore be formulated as an zero-one integer linear program (LP) [2,3]. The complexity of the LP, $O(N2^N)$ [2], increases faster than exponentially. This obstacle appears even for relatively small window sizes. For illustration, the LP for 4x4 window contains over a million variables and constraints. The suboptimal design procedure presented in [4] does not require the use of LP. The only computation involved is data comparisons whose number increases slower than exponentially as a function of the filter window width, thus this optimization algorithm is more efficient and suitable for large scale optimization problems in image and signal processing.

3.2 The adaptive stack filtering

This approach eliminates the stationarity assumption, it is computationally more efficient and it does not require direct computation of the statistics of the corrupted and desired images [5]. It does, however, require the existence of training sequence.

The truth table representation of the PBF has been used in mentioned optimal SF's algorithms. However, the exponential increase of the elements in the truth table of PBF with respect to the number of its variables limits these optimal stack filtering algorithms to be used only for small windows. In order to overcome the above difficulties it is essential to use a concise representation of PBF different from the truth tables. An efficient way to represent PBF is neural network (NN) representation of an arbitrary PBF [6].

3.3 Neural network stack filters

Neural network representation of PBF has several distinct advantages [6]. First, the number of the parameters to describe a SF is much smaller than the number of elements in the truth table of the corresponding PBF. Second, SF's can be implemented using sorting operations in the real domain rather than using the threshold decomposition architecture. This property is useful in software realization of SF's. Third, the relationship between SF's and NN opens up the possibility to find optimal SF's under mean square error (MSE) criterion. This is impossible if the algorithm is based on the truth table representation of PBF. A NN consists of a large number of simple processors called neurones. The neuron has inputs $x_1, x_2, ..., x$. These inputs can come from other units, or from some external source. The inputs $\{x_j\}$ and weights $\mathbf{W} = \{W_j\}$ are accumulated into an activation potential

$$h = \sum_{j=1}^{L} W_j x_j - W_0 \qquad (5)$$

where W_0 denotes the threshold of neuron. The output of the neuron is in general nonlinear function $\sigma(h)$ of the activation potential. It has been shown, that one neuron can realize small subset of Boolean functions known as linearly separable Boolean functions [6]. An arbitrary PBF can be represented by two layer perceptron (one hidden layer) with a sufficient number of neurons in the hidden layer. The optimization problem can be solved using the modified backpropagation algorithm which is summarized for the proposed structure of SF using one neuron as follows [6]:

- 1. Initially set all weights to small positive values
- 2. Present an input vector $\mathbf{X}_N(n)$ and the output $S(n)$
- 3. Use

$$\hat{s}^m(n) = U_S(\mathbf{W}^T(n-1)\mathbf{x}^m(n)) \qquad m = 1, 2, ...M-1$$

- 4. Adjusts the weights by using the recursive algorithm

$$W_0(n) = P\left[W_0(n-1) - \mu \sum_{m=1}^{M-1} \sigma^m(n) \right]$$

$$W_j(n) = P\left[W_j(n-1) + \mu \sum_{m=1}^{M-1} \sigma^m(n) x_j(n) \right]$$

$$j = 1, 2, \ldots, N$$

where

$$\sigma^m(n) = (s^m(n) - \hat{s}^m(n))\hat{s}^m(n)(1 - \hat{s}^m(n))$$

$\mu > 0$ is adaptive step-size

$$P[x] = \begin{cases} x & if \quad x \geq 0 \\ 0 & if \quad x < 0 \end{cases}$$

$$U_s(h) = 1/(1 + \exp(-h))$$

In the case of 2D data (e.g. images) we can optimize this algorithm because the process $X(n)$ has a special structure [8].

4. Realizations of Stack Filters

4.1 Parallel realization

For an M-valued signal, we can first decompose it into a set of binary signals by TD, then filter each binary signal with a binary filter, and finally, add up the results of these operations. This is the so-called threshold decomposition architecture realization or binary processing approach expressed by (2). The binary processing approach can be used in highly parallel VLSI implementation. The binary filters on each level of the TD architecture can be implemented by digital or analog hardware.

The main drawback of this realization is an A/D converter array which perform TD and must precede threshold decomposition architecture. The hybrid realization without A/D converter array is proposed in the following section.

4.2 Hybrid realization

Hybrid realization of SF is based on the stacking property and on the same principle like mixed analog digital realization of a median filter. In the hybrid realization of SF (HRSF) [9], the signal processing is performed on the border between the analog and the digital worlds, where a set of analog signals is used as inputs, and digital data are obtained as the output. The advantage of HRSF is possibility to obtain both analog and digital outputs.

4.3 Realization in the real domain

The positivity of the weights of neural stack filter (NSF) opens up the possibility to calculate the output of the SF in the real domain (without the TD) using sorting operation [6]. The output of the SF over samples $X_1, X_2, \ldots, X_N$ can be calculated as follows. Starting from the higher end of the sorted set of samples, add up the corresponding weights until the sum $W \geq W_0$. The output of the SF is the sample corresponding to the last weight.

5. Simulation Results

To visualize the differences in performance for some of the filters described in this paper and median filter which is widely used in practice we present the results of restoration of images corrupted by impulsive noise. We use common image "Bridge" with raster 256x256 pixels and 8 bit resolution (Fig.3a). The observed image was corrupted by impulsive noise with probability of impulses 10%. In this simulation, the impulses were set to random values from 0 to 255 (Fig.3b). NSF was trained by using the upper left quarter of the original image "Bridge" and noisy image "Bridge". We used the simplest case of NN a single neuron, which gives comparable results like NN with one hidden layer. Training a 3x3 NSF takes less then one minute of CPU on PC AT 386/33 by using optimized algorithm [8]. The filter resulting from the training phase were then used to filter the noisy image "Bridge" (Fig.3d). The same noisy image was filtered by a 3x3 median filter (Fig.3c) and 3x3 optimal stack filter (OSF), that was designed using method in [4]. The MSE and MAE values of the restored images are listed in Tab.1 for image "Bridge" with different probabilities of impulses filtered

Window 3x3	Bridge 0%		Bridge 5%		Bridge 10%		Bridge 15%	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MAE
OSF	2.09	29.90	2.80	62.80	3.67	107.00	4.57	155.00

Tab.1
The MSE and MAE values of the restored image "Bridge"

with optimal filters for 10% probability of impulses and median filter.

The same experiments was performed on image

Window 3x3	Lena 0%		Lena 5%		Lena 10%		Lena 15%	
	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MAE
OSF	0.70	5.25	1.12	20.70	1.77	58.30	2.70	57.00
NSF	0.66	4.92	1.09	***	1.78	***	2.86	***

Tab.2
The MSE and MAE values of the restored image "Lena"

"Lena", which represent image with different statistics. The results are summarized in Tab.2.

From the experimental results one can observe that the NSF gives much better performance than a 3x3

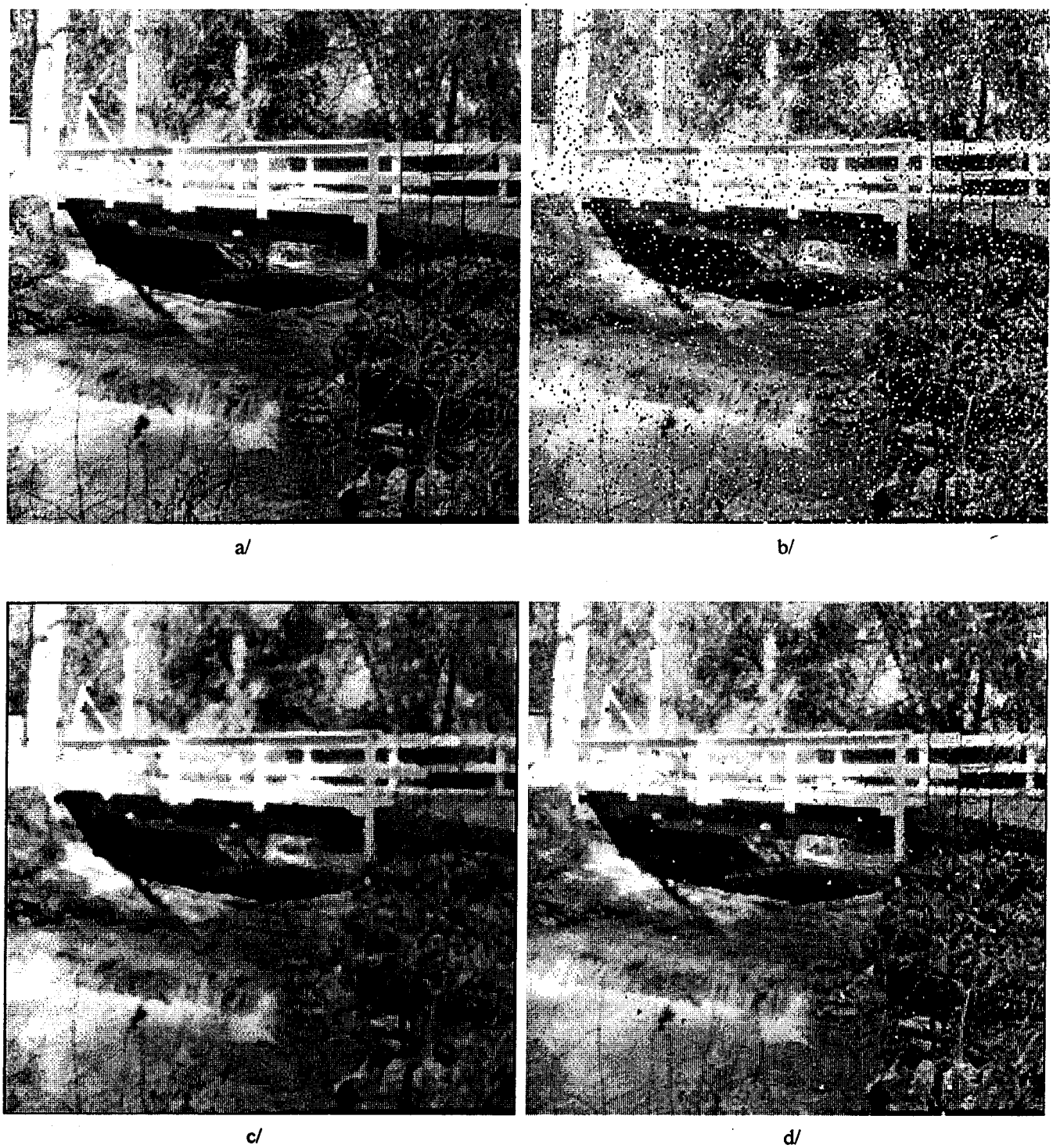

Fig.3
Filtering the image "Bridge" degraded by impulsive noise with 3x3 filters : a/ original image, b/ noisy image, c/ median filter, d/ neural stack filter

median filter and comparable with 3x3 OSF. The filtering of image "Lena" was performed in order to show the robustness of SF against image and noise variations. Therefore, in practice we can train using typical images and noise, and then apply the resulting NSF to a variety of images. In order to investigate the coefficient sensitivity of NSF, the effect of word length changes was simulated. It was shown, that NSF with 5-bits word length gives the same results as the NSF with floating point representation [10]. Such a degree of accuracy might be easily attained with analogue circuitry. This fact opens up a new possibility of using nonlinear digital filters for real time signal processing based on analog hardware.

6. Conclusions

In this paper we describe a large class of non-linear filters, the SF's, and the methods for optimal SF design. Simulation results confirm that at the expense of additional computations we can find filters which are robust to changes in statistical characteristics of image and impulsive noise and are better than conventional median filters. Especially, the NSF as a concise representation of SF can provide novel filter structures which allow to implement SF's effectively without the threshold decomposition and opens up the possibility to use the existing training algorithms of NN optimized for 2D images to estimate optimal SF's. It was shown that NSF based on single neuron gives comparable results as optimal SF based on truth table representation.

7. References

[1] Pitas,I.- Venetsanopoulos, A.N.: Order Statistics in Digital Image Processing. Proceedings of the IEEE, 1992, Vol.80, No.12, pp.1893-1921.

[2] Coyle,E.J.-Lin,J.H.: Stack Filters and the Mean Absolute Error Criterion. IEEE Trans. on ASSP, 1988, Vol.36, No.8, pp.1244-1254.

[3] Coyle, E.J.-Lin,J.H.-Gabbouj,M.:Optimal Stack Filtering and the Estimation and Structural Approaches to Image Processing. IEEE Trans. on ASSP, 1989, Vol.37, No.12, pp.2037-2066.

[4] Zeng,B.- Gabbouj,M.- Neuvo,Y.: A Unified Design Method for Rank Order, Stack, and Generalized Stack Filters Based on Classical Bayes Decision. IEEE Trans.on Circuits and Systems, 1991, Vol.38, No.9, p.1003-1020.

[5] Lin,J.H.- Selike,T.M.- Coyle,E.J.: Adaptive Stack Filtering under the Mean Absolute Error Criterion. IEEE Trans. on ASSP, 1990, Vol.38, No.6, pp.938-954.

[6] Yin,L.- Astola,J.- Neuvo,Y.: A New Class of Nonlinear Filters-Neural Filters. IEEE Trans. on Signal Processing. 1993, Vol.41, No.3, pp.1201-1222.

[7] Marchevský,S.-Drutarovský,M.: Realizácia a návrh kompozičných filtrov pomocou viacvrstvovej neurónovej siete. Zborník prednášok z celoštátneho seminára "Číslicové spracovanie signálov '92", február 1992, Košice, s.108-112 (in Slovak).

[8] Drutarovský,M. A Fast Algorithm for the Design of Weighted Order Statistic Filters. Zborník z konferencie "Image Processing and Neural Networks, máj 1993, Liptovský Mikuláš, s.249-254.

[9] Drutarovský, M.: Hybrid Realization of Stack Filters. Zborník z konferencie "Nové smery v spracovaní signálov II". Liptovský Mikuláš, máj 1994, s. 107 - 110.

[10] Drutarovský, M.: Vplyv dĺžky kódového slova na vlastnosti neurónových kompozičných filtrov. Zborník z 9. Medzinárodnej vedeckej konferencie, Žilina, september 1993, s. 51 - 56 (in Slovak).

About authors,...

Miloš Drutarovský received the M.S. degree in electrical engineering from the Faculty of Electrical Engineering, Technical University in Košice, in 1988. At present he is an assistant professor at the Department of Radioelectronics, of the FEI TU in Košice.

Stanislav Marchevský received the M.S. degree in electrical engineering from the Faculty of Electrical Engineering, Czech Technical University in Prague, in 1976 and Ph.D. degree in radioelectronics from Technical University in Košice in 1985. From 1987 he is the associate professor at the Department of Radioelectronics, of the FEI TU in Košice.