

References

- 1 LUPAS, L., and VERDU, S.: 'Linear multiuser detectors for synchronous code-division multiple-access channel', *IEEE Trans.*, 1989, **IT-35**, (1), pp. 123-136
- 2 MIDDLETON, D.: 'Man-made noise in urban environments and transportation systems: Models and measurements', *IEEE Trans.*, 1973, **COM-21**, pp. 1232-1241
- 3 NIKIAS, C.L., and SHAO, M.: 'Signal processing with alpha-stable distributions and applications' (John Wiley and Sons, New York, 1995)
- 4 GONZALEZ, J.G.: 'Robust techniques for wireless communications in non-Gaussian environments'. PhD Thesis, Department of Electrical and Computer Engineering, University of Delaware, Newark, Delaware, USA, December 1997

Programmable hardware system for class of nonlinear order statistics rational hybrid filters

L. Khrijji, G. Bernacchia, M. Gabbouj and G. Sicuranza

A programmable hardware system developed for a recent class of nonlinear hybrid filters called order statistics-rational hybrid filters (OSRHFs) is presented. The application at hand is noise filtering in grey level images. The proposed hardware system exploits the features of the dynamic logic families which can achieve high speed and compactness, while keeping the power dissipation very low.

Introduction: Nonlinear filters have been found effective in removing different types of noise from images. This is especially the case if the noise corrupting the image is not Gaussian and/or image details are not to be blurred. For this purpose, a number of order statistics based techniques have been proposed [1].

In this Letter we describe an extension to our work in [2], where we first proposed the hybrid filter structure. Here, we present the results of a study into the performance of these new nonlinear rational type hybrid filters and describe a programmable hardware system for implementing them. Three order statistics subfilters are first computed in the first layer of the hybrid structure. The idea here is to remove impulsive type noise while causing minimal distortion in the image. Order statistics have been found to be very effective in such applications. The final layer of OSRH filters is a rational function. The aim of the final layer, in addition to its detail preserving capability [3], is to attenuate Gaussian type noise. The hybrid approach is therefore suitable for applications where both impulsive and Gaussian noise are expected.

Order statistics-rational hybrid filters: The output of an order statistics-hybrid rational filter (OSRHF), is the result of a rational function operating on three order statistics subfilters $\{\Phi_1, \Phi_2, \Phi_3\}$ and is given by

$$y(n) = \Phi_2(n) + \frac{\sum_{i=1}^3 \alpha_i \Phi_i(n)}{h + k(\Phi_1(n) - \Phi_3(n))^p} \quad (1)$$

The set of coefficients $\alpha = [\alpha_1, \alpha_2, \alpha_3]^T$ is used as weights for the subfilter outputs, while parameters p , h , and k are some positive constants. Parameter k plays an important role in rational filtering as it is used to gauge the effect of the nonlinear term in the denominator.

In this Letter, the centre subfilter (Φ_2) is a centre weighted median; the other two are bidirectional median subfilters. We used $\alpha = [1, -2, 1]^T$ satisfying $\sum_{i=1}^3 \alpha_i = 0$ and $p = 2$. The resulting filter is thus called a median-rational hybrid filter (MRHF), and is shown in Fig. 1a.

Since median filters have well-known efficient software and hardware implementations, we limit here our discussion to the hardware realisation of the rational operation.

Programmable hardware implementation: As can be seen from its structure, the MRHF could be split into two main sections:

(i) a block with the three median filters, which involves sorting the incoming samples from the image; the results of these filters are exact

(ii) a block which realises the rational function and therefore requires some arithmetic function.

Each of these sections presents some problem when it has to be implemented. In fact, the parameters used in the rational operator are not the results of a previous optimisation procedure. A programmable structure allows the user to explore different combinations and find the best tradeoff between noise attenuation and edge preservation among the tested operators. We used the differential cascode voltage switch logic (DCVSL) in the dynamic configuration. Fig. 1b shows the static and dynamic configurations of this family where there is no direct path from ground to the power supply but for a short time during the switching. For our implementation we used the standard binary representation for median filters and the residue number system (RNS) [4] for the rational function.

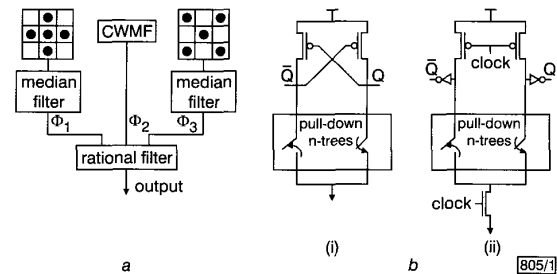


Fig. 1 Structure of MRHF and basic configurations of differential cascode voltage switch logic

- a MRHF structure
b Basic logic configurations
(i) static configuration
(ii) dynamic configuration

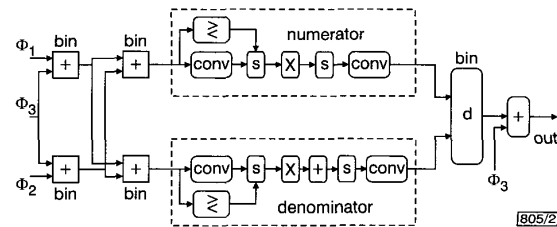


Fig. 2 Block diagram of rational function implementation

s: scale, conv.: converter, d: divider, ϕ_i : output of i th median filter

An RNS system is defined by a set of moduli m_1, m_2, \dots, m_n . An integer a can be represented in this system by means of the residues modulo m_i : $a = a_1, a_2, \dots, a_n$ where $a_i = a \bmod m_i$. For this specific application we used three modules: $m_1 = 15$, $m_2 = 31$ and $m_3 = 32$, therefore we can represent integers in the range $[-7440, 7440]$. Each module requires just three adders and the result is exact. The result of the rational operator in eqn. 1 uniquely depends on the sign of the numerator; we then simply store this sign bit and perform all the operations with unsigned (positive) numbers. This solution enables us to double the dynamic range and then to reduce the need for scaling. Fig. 2 shows the block diagram of the circuitry for the rational function. To calculate the numerator and denominator of eqn. 1, we developed a simple scaling algorithm for converting the RNS representation into the so called mixed radix system (MRS) representation. Given the set of modules $\{m_i\}_{i=1, \dots, n}$ an integer a can be represented in the MRS as

$$a = \alpha_1 + \alpha_2 m_1 + \alpha_3 m_1 m_2 = (\alpha_1, \alpha_2, \alpha_3) \quad (2)$$

where α_i can be directly obtained from the residues m_i . If we scale the number a by the module m_1 , we can write: $a' = a/m_1 = \alpha_1/m_1 + \alpha_2 + \alpha_3 m_2$. Since $\alpha_1/m_1 < 1$. Exploiting the features of the used set of modules, the residues of the scaled integer a' can be calculated as: $a'_1 = (r + \alpha_2 + \alpha_3) \bmod m_1$, $a'_2 = (r + \alpha_2) \bmod m_2$, $a'_3 = (r + \alpha_2 - \alpha_3) \bmod m_3$. The parameter r is used to reduce the level of truncation errors introduced by the scaling ($r =$

0, 1 depending on $\alpha_1 < m_1/2$ or $\alpha_1 \geq m_1/2$). The final conversion from RNS to binary is obtained as sum of products such as eqn. 2 and the output is represented by 13 bits.

Experimental results: To assess the performance of the proposed programmable hardware system we used different images mainly from the TUT image database [5]. Three corrupted test images were derived from the Bridge image in [5], by adding to the original image i.i.d. noise having the following contaminated Gaussian probability distribution, $v \approx (1 - \lambda)N(0, \sigma_n) + \lambda N(0, \sigma_n/\lambda)$. Three values of λ were chosen 0.1, 0.2 and 1, and one set of images was formed with $SNR = 3$ dB. For objective comparison, MAE and MSE criteria were used (Table 1). These (Imp) were compared with the theoretical MRHF (Th), and the well known techniques, stack filters (Stack) and the rank-order morphological filters (ROMFs).

From Table 1 we can conclude that in most cases the implemented structure gives even better results than the theoretical structure due to the effects of rounding errors in the approximation of the rational function. The use of high speed and low power logic families enables us to obtain very good performances. The speed requirements for real time applications are variable depending on the image size and on the frame rate. For 1024×768 images at a frame rate of 50Hz a frequency of at least 40 MHz is required in order to perform all operations. The results obtained with the DCVS logic show that the rise and fall times for a complex gate are $\sim 1-1.5$ ns. For instance, the gate performing the calculation of the carry chain has a rise time of 0.938 ns and a fall time of 1.377 ns. Such characteristics make it possible to satisfy real-time processing requirements.

Table 1: Quantitative comparison

λ	3dB	Noisy	ROMF	Stack	Th	Imp
0.1	MAE	23.445	11.400	11.402	10.465	10.154
	MSE	1503.50	240.25	227.01	196.92	186.01
0.2	MAE	25.172	11.400	11.400	10.920	10.600
	MSE	1508.10	240.25	227.01	213.81	202.29
1	MAE	31.002	13.540	16.238	13.380	13.030
	MSE	1498.80	320.50	429.21	300.10	286.28

Conclusions: In this Letter we have introduced order statistics-rational hybrid filters. A programmable hardware system has been proposed and evaluated. The implementation uses the standard binary representation for median subfilters and the RNS for the rational operator. A dynamic logic based on the DCVS logic has been chosen according to the characteristics of low power consumption and high processing speed. Our tests have shown that the implemented system is very robust and that its features make it suitable for real-time applications.

© IEE 2000

13 March 2000

Electronics Letters Online No: 20000665

DOI: 10.1049/el:20000665

L. Khriji and M. Gabbouj (Signal Processing Lab., TUT, PO Box 553, FIN-33101 Tampere, Finland)

E-mail: lazhar@cs.tut.fi

G. Bernacchia and G. Sicuranza (D.E.E.I., University of Trieste, via Valerio, 10, 34100 Trieste, Italy)

References

- PITAS, I., and VENETSANOPOULOS, A.N.: 'Nonlinear digital filters: Principles and applications' (Kluwer Academic, Dordrecht, Holland, 1991)
- KHRIJI, L., and GABBOUJ, M.: 'Median-rational hybrid filters for image restoration', *Electron. Lett.*, 1998, **34**, (10), pp. 977-979
- RAMONI, G.: 'The rational filter for image smoothing', *IEEE Sig. Process. Lett.*, 1996, **3**, (3), pp. 63-65
- DUGDALE, M.: 'VLSI implementation of residue adders based on binary adders', *IEEE Trans.*, 1992, **CASII-39**, (5), pp. 325-329
- GABBOUJ, M., and TABUS, I.: 'TUT noisy image database'. Technical Report, No. 13, Tampere University of Technology, December 1994

Vessel radiated noise recognition with fractal features

Su Yang, Zhishun Li and Xinlong Wang

Vessel radiated noise is somewhat similar to fractional Brownian motion. Accordingly, the standard deviations of differential sequences and fractal dimensions are extracted as features that reflect time-domain information other than traditional features. Six types of vessel can be classified correctly, which indicates that fractal features are useful for vessel noise recognition.

Introduction: Vessel radiated noise is an important information source which reflects some physical characteristic of a vessel. In recognising it, the traditional method focuses on features in the frequency domain such as the line-component spectrum [1, 2]. Meanwhile, features in the time domain which are also helpful have, however, not yet attracted much attention. There are some difficulties for traditional features. A practical problem is how to separate the line-component spectrum from the continuous spectrum accurately. Generally, if more effective features are used in the classification, then a higher recognition accuracy can be achieved. Therefore, novel features that can carry complementary information with regard to traditional features are in demand. Recently, it was found that vessel noise is somewhat similar to fractional Brownian motion (fBm) [3], the differential sequence of which satisfies the T^H law [4]: its power spectrum is like $1/f$ noise and its behaviour can be described by the fractal dimension [5, 6]. Thus, standard deviations of the differential sequences and fractal dimensions are taken as features here. In calculating the fractal dimensions, the sectionalising method is used to obtain robust features. By only using a linear classifier, samples of six classes can be correctly classified.

Fractional Brownian motion feature: Recently, it has been reported that vessel noise has some character like fractional Brownian motion [3], the differential sequence of which satisfies the T^H law [4]:

$$\text{var}[B_H(t+T) - B_H(t)] = C_H T^H$$

where $B_H(t)$ denotes fBm with parameter H , and C_H is usually deemed as constant when H is given. For a sequence of vessel noise s_i ; $i = 1, 2, \dots, N$, when the step length is k , the standard deviation is

$$\sigma_k = \sqrt{\sum_{i=1}^{N-k} (s_{i+k} - s_i - \mu_k)^2 / (N-k)}$$

where μ_k is the mean of the differential sequence $s_{j+k} - s_j$; $j = 1, 2, \dots, N-k$. According to the T^H law, the relationship between σ_k with respect to k reflected in a log-log plot will be a straight line. Then H can be acquired by least-squares fitting, which is used as the feature in [3]. However, vessel noise could be approximated by, but never be given accurately, by fBm in practice. This means that the relationship between $\log \sigma_k$ and $\log k$ is not as linear as for the ideal situation, which will result in high random fitting errors. Such errors being harmful to the stability of features should be avoided in any case. We therefore use the standard deviation of the differential sequence at some given step length as the feature because it is more stable than the fitting result of H in the case where the actual model slightly deviates from fBm.

Fractal dimension feature: The fractal dimension is of high importance in describing the behaviour of fBm. We use the blanket-covering dimension as the feature, for it has the property that the computation of the fractal dimension is invariant to signal shifting in time and amplitude co-ordinates [6]. For a one dimensional digital signal $f(n)$; $n = 0, 1, \dots, N$, choose a scale r to form an upper envelop $U_r(n)$ and a lower envelop $L_r(n)$, where

$$U_r(n) = \max\{U_{r-1}(n-1), U_{r-1}(n) + 1, U_{r-1}(n+1)\}$$

$$L_r(n) = \min\{L_{r-1}(n-1), L_{r-1}(n) - 1, L_{r-1}(n+1)\}$$

$$U_0(n) = L_0(n) = f(n)$$

Then, calculate the fractal measurement