INFORMATICA, 2002, Vol. 13, No. 1, 23–36

Layered Polynomial Filter Structures

Kazys KAZLAUSKAS, Jaunius KAZLAUSKAS
Institute of Mathematics and Informatics
Vilnius Pedagogical University
Akademijos 4, 2600 Vilnius, Lithuania
e-mail: kazlausk@ktl.mii.lt

Received: September 2001

Abstract. It is shown that nonlinear Volterra, polynomial autoregressive, and bilinear filters have the same layered implementation procedure. Using the layered structure, the order of nonlinearity can be increased by adding more layers to the structure. The structure is modular and consists of the simple moving average (MA) or autoregressive (AR) filters which can be added to the structure to achieve a desired degree of complexity. In addition, the modular layered structures admit very large scale integration (VLSI) implementation of the polynomial nonlinear filters.

Key words: polynomial filters, layered structures.

1. Introduction

Recent years have witnessed a considerable activity in nonlinear modelling (Pitas and Venetsanopoulos, 1990), (Dokic and Clarkson, 1993). The reason for such an interest is that the most natural systems are nonlinear (Celka, Bershad, and Vesin, 2001). A variety of nonlinear filtering algorithms have been developed. New techniques continue to develop and this field expands both in the theoretical and application directions. Nonlinearities have been viewed as deviations from the linear behavior. In many approaches locally-linear or piecewise-linear approximations are used to quantify the non-linear behavior. Unlike linear filters which are completely characterized by the impulse response, it is difficult to find a unified framework for describing nonlinear filters.

There is a great variety of nonlinear system structures. The first task is to determine the most appropriate structure for the underlying problem. Good reviews of nonlinear system structures can be found (Billings, 1980; Ljung, 1987; Schetzen, 1980). Among the nonlinear filters, polynomial filters, i.e., Volterra, polynomial autoregressive, and bilinear filters have been one of the most widely used nonlinear system representation (Billings, 1980). One of the problems related to the polynomial filter representation is the computational complexity. As the polynomial order and/or memory increases, the number of parameters in the polynomial filter increases rapidly (Raz and Van Veen, 1998). Existing implementations of polynomial filters are computationally expensive. It is difficult to change the order of the nonlinearity.

There are two main approaches for reducing the Volterra filter complexity (Raz and Van Veen, 1998). The Volterra filter is approximated using a cascade structure composed

of linear filters in series with memoryless nonlinearities (Frank, 1994; Korenberg, 1991). Another approach is the tensor basis approximation, which represents the Volterra kernel as a linear combination of tensor products of basis vectors (Chiang, Nikias, and Venetsanopoulos, 1986; Panicker and Mathews, 1996). Priestley (1988) proposed to represent a nonlinear time series as linear models with variable parameters. Priestley analyzed exponential AR, threshold AR, and bilinear models and showed that they were special cases of the state dependent models.

This paper develops a unified, computationally efficient, and modular structures for polynomial nonlinear filter implementation using the layered approach. The layered approach is related to Priestley's state dependent linear models (Priestley, 1988).

2. Overview of Polynomial Filter Implementation

Volterra filters can be realized as linear filters by changing the nonlinear problem into a linear one. For example, the second order Volterra filter

$$y(n) = \sum_{i_1=0}^{N} a(i_1)u(n-i_1) + \sum_{i_1=0}^{N}\sum_{i_2=0}^{N} a(i_1,i_2)u(n-i_1)u(n-i_2) \tag{1}$$

can be realized as follows:

$$y(n) = A^T u, \tag{2}$$

where

$$u = \left[u(n)u(n-1), ..., u(n-N), u^2(n), u(n)u(n-1), ..., u^2(n-N)\right]^T, \tag{3}$$

$$A = \left[a(0), a(1), ..., a(N), a(00), a(01), ..., a(NN)\right]^T. \tag{4}$$

The output is linear and quadratic combinations of the input. The coefficients of the filter form a vector A. The pth order Volterra filter implementation requires N^p multiplications for the formation of the input vector, and $\mathrm{O}(N^p)$ computations for linear filtering. The nonlinear processor forms the input vector u. The input autocorrelation matrix has a large eigenvalue spread and the convergence of such linear mean square adaptive filters is slow (Mathews and Lee, 1988). For the slow convergence reason, Volterra filters have not been extensively applied.

The most important requirements for VLSI design are low computational load, local connectivity, and modularity. Although Volterra filters are implemented as linear filters, the obtained structures are not attractive for the VLSI implementation. The input length increases with an increase of the filter order. The adaptation is global and has to be connected to all inputs and all weights. The problem of connectivity cannot be solved by systolic implementation.

The Volterra filter can also be implemented as a multichannel MA linear filters. Perry and Parker (1980) used a nonlinear combination of the input. A second order Volterra filter can be represented as follows:

$$y(n) = \sum_{i_1=0}^{N} a(i_1)u(n-i_1) + \sum_{i_2=0}^{N-1} \left(\sum_{i_1=i_2}^{N+i_2-1} a(i_1,i_2)v(n-i_1,i_2) \right), \tag{5}$$

where

$$v(n-i_1,i_2) = u(n-i_1)u(n-i_1-i_2). \tag{6}$$

The second order Volterra filter can be implemented as the sum of the outputs of $N+1$ linear filters. A nonlinear filter is implemented by a linear filter and a nonlinear processor. A variety of algorithms have been developed based on this approach. However, lattice structures based on the multichannel approach give over-parametrized model.

Other implementation of the second order Volterra filter has been presented using a matrix representation for the second order term (Chiang *et al.*, 1986). The second order filter output can be written as

$$y(n) = Y^T(n)VY(n). \tag{7}$$

The matrix V can be decomposed in different ways. The matrix V is decomposed into a sum of matrices as follows:

$$y(n) = \sum_{i=1}^{N} \varepsilon_i \left[Y^T(n)Q_i\right] \left[Q_i^T Y(n)\right]. \tag{8}$$

The second order filter can be realized by using N linear filters each with the output $Q_i^T Y(n)$. Other realizations are achieved by coupling the matrix decomposition approach with distributed arithmetic. These realizations are efficient only for second order filters.

Nonlinearity can also be modelled by introducing static nonlinearities at the output of linear filters. Depending on the type of nonlinearity, the model can approximate a nonlinear dynamic by realizing a Volterra series. Neural networks are examples of such implementations where a linear combiner is followed by a nonlinearity. This type of filter structure is restricted in degrees of freedom and cannot implement an arbitrary nonlinear function.

The best advantage of polynomial filters is their systematic approach toward arbitrary nonlinearities. The higher order terms can be included in the model according to the required accuracy. Another advantage is their generalized structure which includes linear filters, making polynomial filters a natural extension of linear filters. The disadvantages of these filters are over-parametrization and difficult implementation procedures. The aim of the paper is to remove these negatives by finding a computationally efficient implementation.

3. Layered Volterra Filter

The output $y(n)$ of any causal, discrete-time, time-invariant pth order N length nonlinear filter can be represented as a function of the input sequence $u(n)$ (Raz and Van Veen, 1998)

$$y(n) = a'(0) + \sum_{i_1=0}^{N-1} a(i_1)u(n-i_1) + \sum_{i_1=0}^{N-1}\sum_{i_2=0}^{N-1} a(i_1,i_2)u(n-i_1)u(n-i_2) + \cdots + \sum_{i_1=0}^{N-1}\dots\sum_{i_p=0}^{N-1} a(i_1,\dots,i_p)u(n-i_1)\dots u(n-i_p). \tag{9}$$

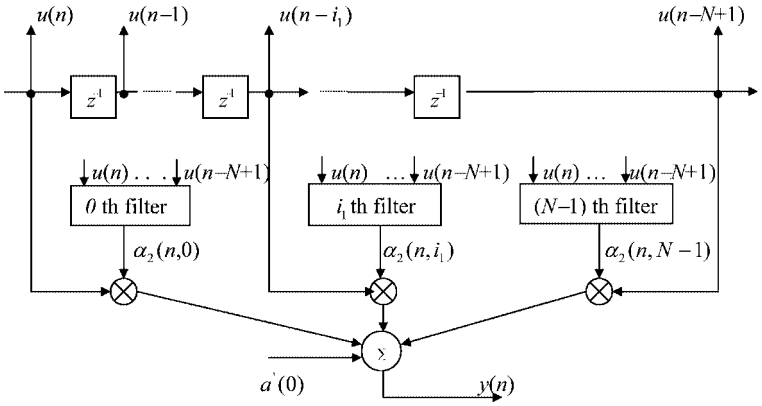
Rearrange (9) as follows:

$$y(n) = a'(0) + \sum_{i_1=0}^{N-1}\Bigg(a(i_1) + \sum_{i_2=0}^{N-1}\Bigg(a(i_1,i_2) + \sum_{i_3=0}^{N-1}\Big(a(i_1,i_2,i_3) + \cdots + \sum_{i_{p-1}=0}^{N-1}\big(a(i_1,\dots,i_{p-1}) + \sum_{i_p=0}^{N-1} a(i_1,\dots,i_p)u(n-i_p)\big)u(n-i_{p-1})\Big)\dots\Bigg)u(n-i_2)\Bigg)u(n-i_1). \tag{10}$$

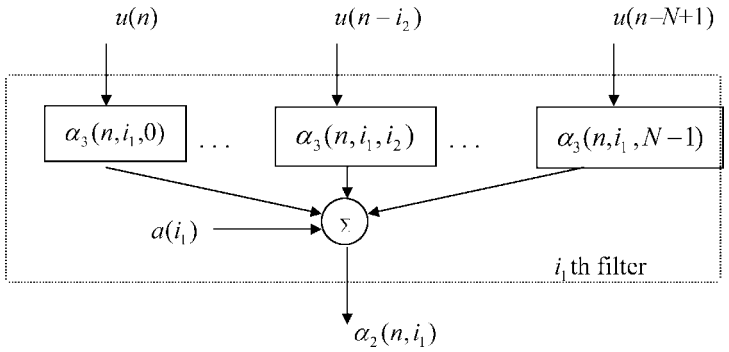
From (10) we obtain a layered representation of the pth order Volterra filter (9):

$$\begin{aligned}
&\text{1st layer: } y(n) = a'(0) + \sum_{i_1=0}^{N-1} \alpha_2(n,i_1)u(n-i_1),\\
&\text{2nd layer: } \alpha_2(n,i_1) = a(i_1) + \sum_{i_2=0}^{N-1} \alpha_3(n,i_1,i_2)u(n-i_2),\\
&\dots\dots\dots\dots\dots\dots\dots\dots\\
&(p-1)\text{st layer: } \alpha_{p-1}(n,i_1,\dots,i_{p-2}) = a(i_1,\dots,i_{p-2}) + \sum_{i_{p-1}=0}^{N-1} \alpha_p(n,i_1,\dots,i_{p-1})u(n-i_{p-1}),\\
&p\text{th layer: } \alpha_p(n,i_1,\dots,i_{p-1}) = a(i_1,\dots,i_{p-1}) + \sum_{i_p=0}^{N-1} a(i_1,\dots,i_p)u(n-i_p).
\end{aligned} \tag{11}$$

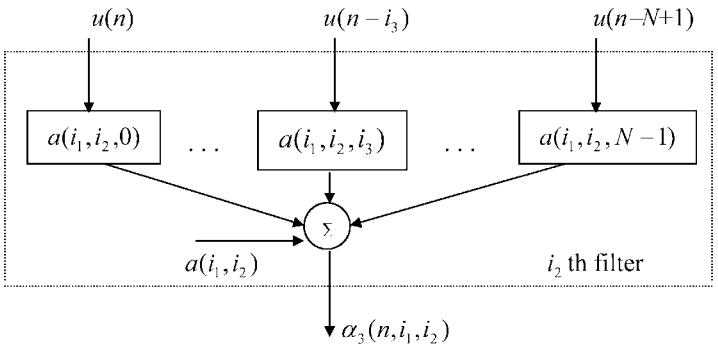
The pth order Volterra filter is realized with p layers of MA filters. The 1st layer is an MA filter with variable coefficients. The coefficients of the 1st layer are modelled with N MA filters of the 2nd layer. The coefficients of the 2nd layer are modelled with N^2 MA filters of the 3rd layer, and so on. The p th layer has N^{p-1} linear MA filters with constant coefficients. The p th order of nonlinearity is described by the $(p-1)$th layer variations.



(a)



(b)



(c)

Fig. 1. The 3rd order layered nonlinear Volterra filter structure.
(a) 1st layer filter; (b) i_1th filter of the 2nd layer; (c) i_2th filter of the 3rd layer.

For exampe, if $p = 3$, then the 3rd order Volterra filter has three layers:

$$\begin{aligned}
&\text{1st layer: } y(n) = a'(0) + \sum_{i_1=0}^{N-1} \alpha_2(n, i_1) u(n - i_1),\\
&\text{2nd layer: } \alpha_2(n, i_1) = a(i_1) + \sum_{i_2=0}^{N-1} \alpha_3(n, i_1, i_2) u(n - i_2),\\
&\text{3rd layer: } \alpha_3(n, i_1, i_2) = a(i_1, i_2) + \sum_{i_3=0}^{N-1} a(i_1, i_2, i_3) u(n - i_3).
\end{aligned} \tag{12}$$

The 3rd order layered nonlinear Volterra filter structure is shown in Fig. 1.

4. Layered Polynomial Autoregressive Filter

Polynomial autoregressive filters can be implemented in a layered manner using linear AR filters. Any pth order N length polynomial autoregressive filter can be implemented in p layers. The jth layer consists of N^{j-1}AR filters.

The pth order N length polynomial autoregressive filter is described as

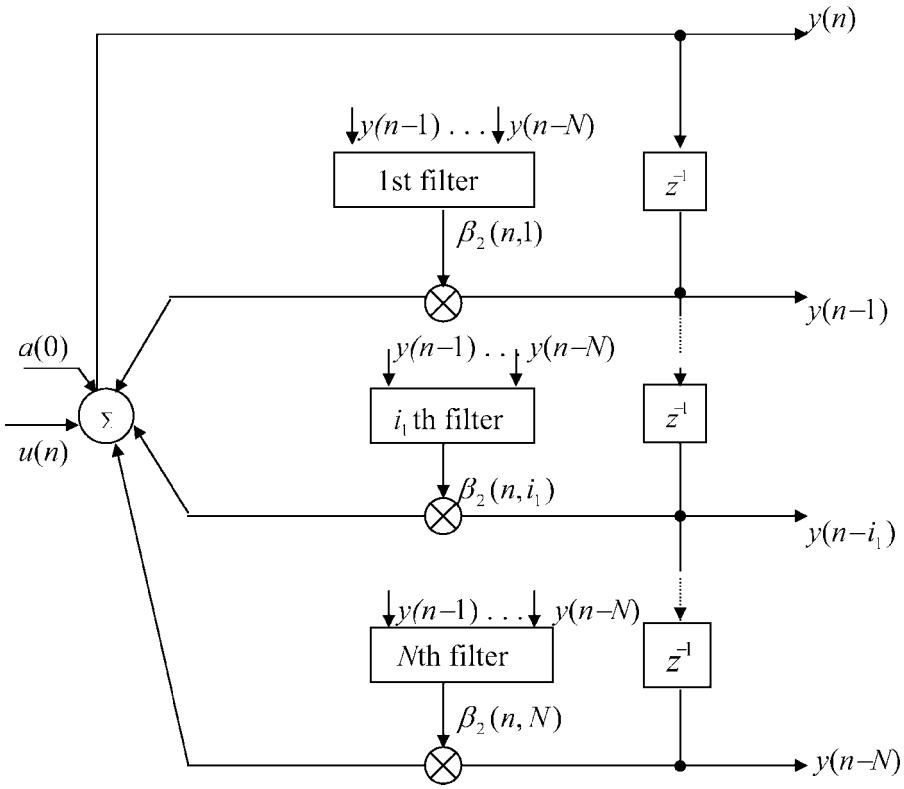
$$\begin{aligned}
y(n) \;=\; & u(n) + a(0) + \sum_{i_1=1}^{N} a(i_1) y(n - i_1) + \sum_{i_1=1}^{N} \sum_{i_2=1}^{N} a(i_1, i_2) y(n - i_1) y(n - i_2)\\
& + \cdots + \sum_{i_1=1}^{N} \cdots \sum_{i_p=1}^{N} a(i_1, \cdots, i_p) y(n - i_1) \dots y(n - i_p).
\end{aligned} \tag{13}$$

We have from (13)

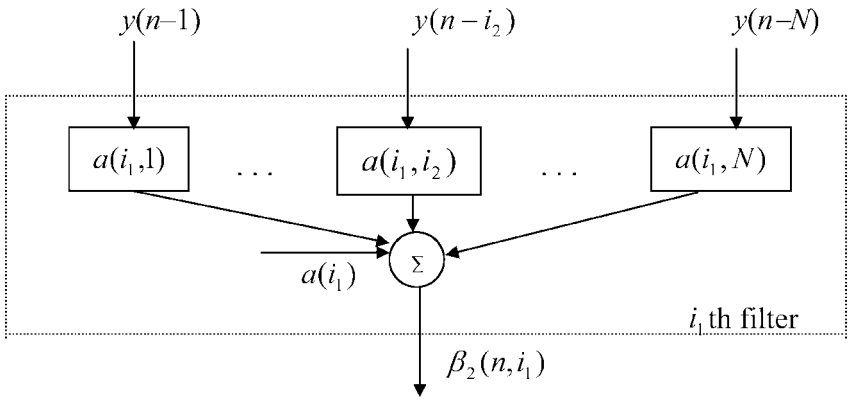
$$\begin{aligned}
y(n) \;=\; & u(n) + a(0) + \sum_{i_1=1}^{N} \Bigg(a(i_1) + \sum_{i_2=1}^{N} \Bigg(a(i_1, i_2) + \cdots\\
& + \sum_{i_{p-2}=1}^{N} \Big(a(i_1, \dots, i_{p-2}) + \sum_{i_{p-1}=1}^{N} \big(a(i_1, \dots, i_{p-1})\\
& + \sum_{i_p=1}^{N} a(i_1, \dots, i_p) y(n - i_p) \big) y(n - i_{p-1}) \Big) \dots \Bigg) y(n - i_2) \Bigg) y(n - i_1).
\end{aligned} \tag{14}$$

Then, from (14) we obtain the following layered representation of the pth order polynomial autoregressive filter (13):

$$\text{1st layer: } y(n) = u(n) + a(0) + \sum_{i_1=1}^{N} \beta_2(n, i_1) y(n - i_1),$$



(a)



(b)

Fig. 2. The 2nd order layered polynomial autoregressive filter
(a) 1st layer filter; (b) i_1th filter of the 2nd layer.

$$
\begin{aligned}
&\text{2nd layer: } \beta_2(n,i_1) = a(i_1) + \sum_{i_2=1}^{N} \beta_3(n,i_1,i_2)y(n-i_2),\\
&\text{.....}\\
&(p-1)\text{st layer: } \beta_{p-1}(n,i_1,...,i_{p-2}) = a(i_1,...,i_{p-2})\\
&\qquad\qquad + \sum_{i_{p-1}=1}^{N} \beta_p(n,i_1,...,i_{p-1})y(n-i_{p-1}),\\
&p\text{th layer: } \beta_p(n,i_1,...,i_{p-1}) = a(i_1,...,i_{p-1}) + \sum_{i_p=1}^{N} a(i_1,...,i_p)y(n-i_p).
\end{aligned}
\tag{15}
$$

As an example, a second order polynomial autoregressive filter has two layers:

$$
\begin{aligned}
&\text{1st layer: } y(n) = u(n) + a(0) + \sum_{i_1=1}^{N} \beta_2(n,i_1)y(n-i_1),\\
&\text{2nd layer: } \beta_2(n,i_1) = a(i_1) + \sum_{i_2=1}^{N} a(i_1,i_2)y(n-i_2).
\end{aligned}
\tag{16}
$$

The 2nd order polynomial autoregressive filter structure is shown in Fig. 2.

5. Layered Bilinear Filter

The discrete-time bilinear filter has the form (Priestley, 1988):

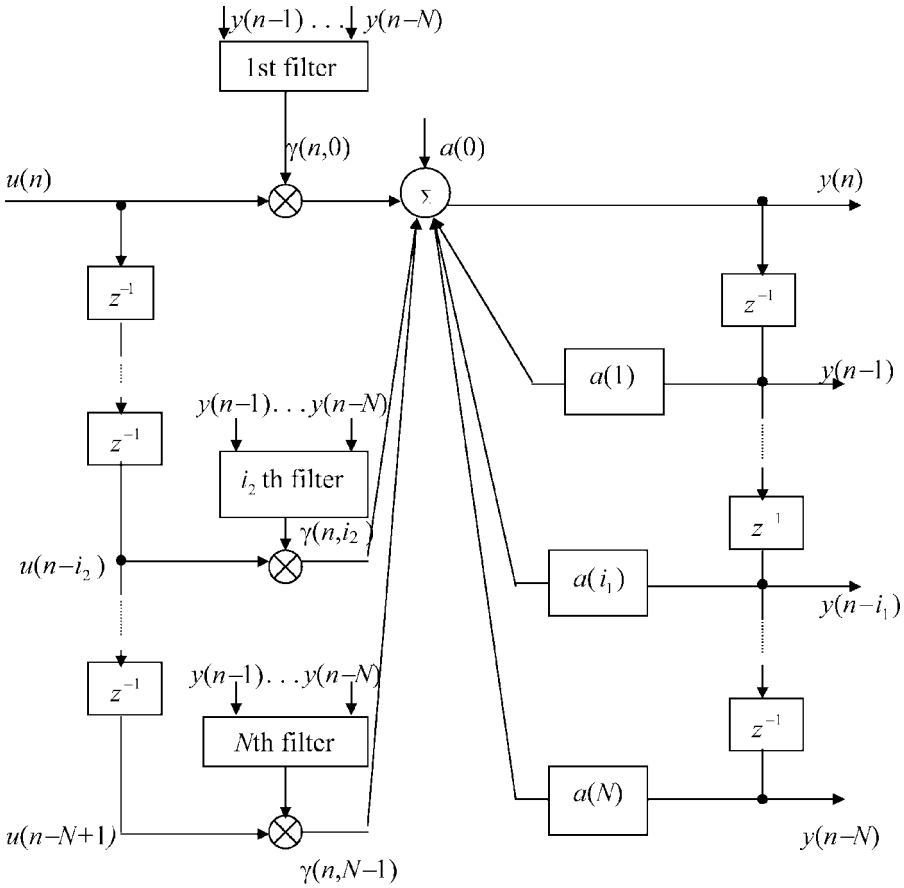
$$
\begin{aligned}
y(n) \;=\; & a(0) + \sum_{i_1=1}^{N} a(i_1)y(n-i_1) + \sum_{i_2=0}^{N-1} c(i_2)u(n-i_2)\\
& + \sum_{i_1=1}^{N}\sum_{i_2=0}^{N-1} b(i_1,i_2)y(n-i_1)u(n-i_2).
\end{aligned}
\tag{17}
$$

We obtain from (17)

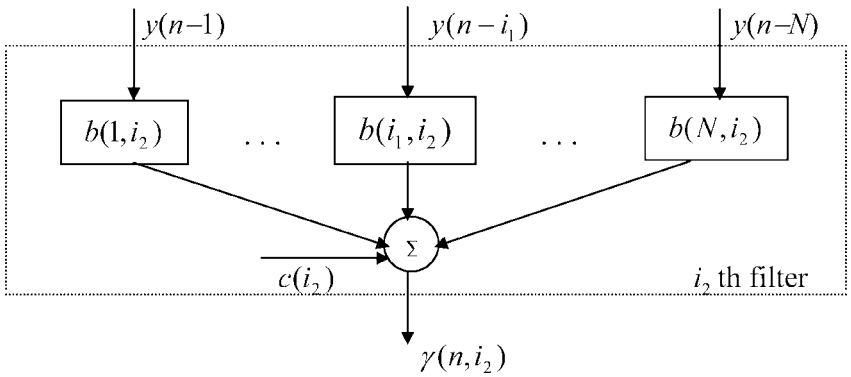
$$
y(n) = a(0) + \sum_{i_1=1}^{N} a(i_1)y(n-i_1) + \sum_{i_2=0}^{N-1}\Big(c(i_2) + \sum_{i_1=1}^{N} b(i_1,i_2)y(n-i_1)\Big)u(n-i_2) \tag{18}
$$

and

$$
y(n) = a(0) + \sum_{i_2=0}^{N-1} c(i_2)u(n-i_2) + \sum_{i_1=1}^{N}\Big(a(i_1) + \sum_{i_2=0}^{N-1} b(i_1,i_2)u(n-i_2)\Big)y(n-i_1). \tag{19}
$$

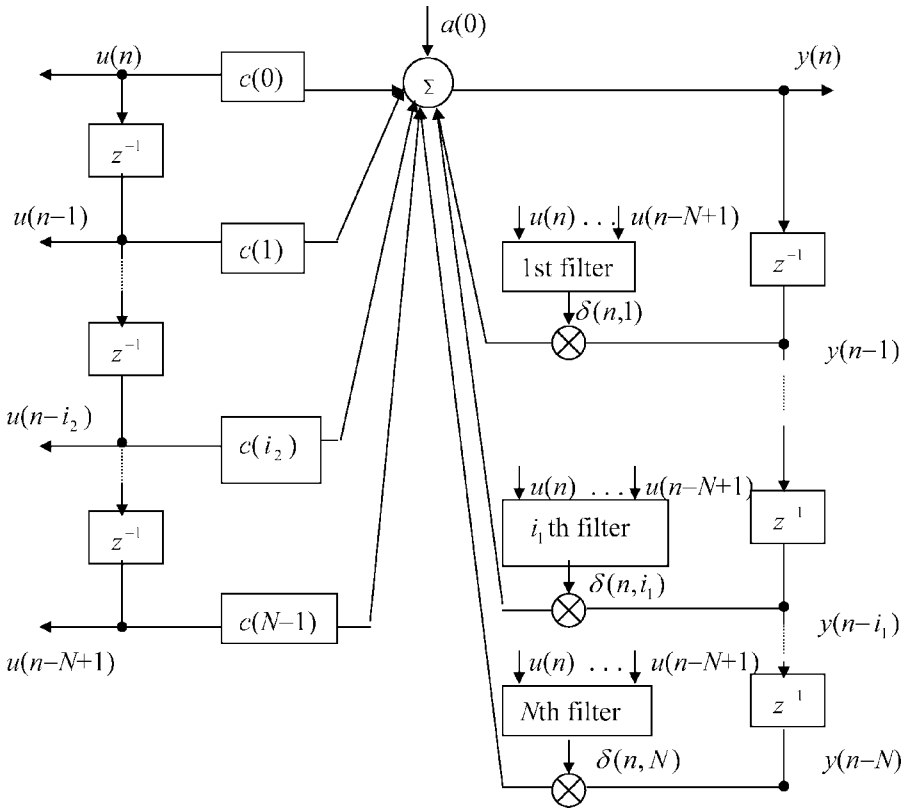


(a)

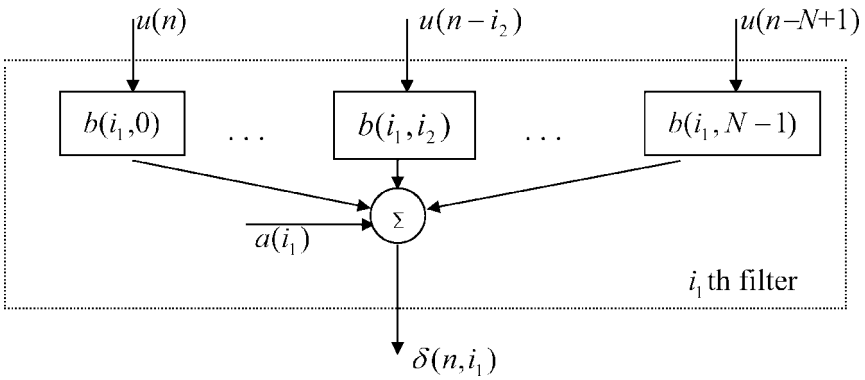


(b)

Fig. 3. First layered bilinear filter structure.
(a) 1st layer filter; (b) i_2th filter of the 2nd layer.



(a)



(b)

Fig. 4. Second layered bilinear filter structure.
(a) 1st layer filter; (b) i_1th filter of the 2nd layer.

Then, we have from (18) the First layered bilinear filter:

$$\begin{aligned}&\text{1st layer: } y(n) = a(0) + \sum_{i_1=1}^{N} a(i_1)y(n-i_1) + \sum_{i_2=0}^{N-1} \gamma(n,i_2)u(n-i_2),\\ &\text{2nd layer: } \gamma(n,i_2) = c(i_2) + \sum_{i_1=1}^{N} b(i_1,i_2)y(n-i_1).\end{aligned} \tag{20}$$

The AR part has a linear dependence, and the MA part has a nonlinear dependence. The MA part has output dependent coefficients.

The First layered bilinear filter structure is shown in Fig. 3.

We obtain from (19) the Second layered bilinear filter:

$$\begin{aligned}&\text{1st layer: } y(n) = a(0) + \sum_{i_2=0}^{N-1} c(i_2)u(n-i_2) + \sum_{i_1=1}^{N} \delta(n,i_1)y(n-i_1),\\ &\text{2nd layer: } \delta(n,i_1) = a(i_1) + \sum_{i_2=0}^{N-1} b(i_1,i_2)u(n-i_2).\end{aligned} \tag{21}$$

The MA part has a linear dependence, and the AR part has a nonlinear dependence. The AR part has input dependent coefficients.

The Second layered bilinear filter structure is shown in Fig. 4.

The number of multiplications per time instant for the bilinear filter (17) is $2N(N+1)$. The number of multiplications per time instant for the First layered bilinear filter and for the Second layered bilinear filter is $N(N+2)$.

Table 1 shows the number of multiplications per one time instant for bilinear filter (17) and for layered bilinear filters (20) and (21).

Table 1

Multiplication complexity comparison for bilinear and layered bilinear filters

Filter length N	Bilinear filter (17)	Layered bilinear filters (20) or (21)
10	220	120
20	840	440
30	1860	960
40	3280	1680
50	5100	2600
100	20200	10200

6. Stability

Variable nature of the layered filter helps us define a dynamical stability for such filters. Stability of the usual filters depends on the location of poles and zeros in the z plane.

We extend it to nonlinear filters using their layered descriptions. Due to the filters with variable parameters in the layered structure, the location of the poles and zeros also vary with time. Therefore, we call such filters to have variable poles and zeros. From this definition we can argue that nonrecursive filters with variable parameters are stable in the sense that, as long as their parameter values are finite, they statisfy bounded-input-bounded-output stability. Each module in the layered structure of such filters will have a bounded output and thus the variable parameter of the above module will remain finite and result in overall nonlinear filter stability.

A recursive filter with variable parameters and its stability analysis is more complicated. A stability can be enforced by limiting the output of the second layer filters such that the variable coefficients of the 1st layer for all instants remain within the unit circle. A more weak condition can be defined by allowing the variable poles for finite time to go outside the unit circle. During their visit to the unstable zone, the variable poles must have an effect on the overall system states such that they diminish in magnitude thus returning within the unit circle.

7. Conclusion

The polynomial nonlinear filters can be efficiently implemented through layered structures. The layered structures of polynomial nonlinear filters are computationally efficient, have a modular structure, and can be implemented using simple moving average or autoregressive linear filters.

References

Billings, S.A. (1980). Identification of nonlinear systems – A survey. *Proc. Inst. Elect. Eng.*, **127**, 272–285.

Celka, P., N. J. Bershad, J. Vesin (2001). Stochastic gradient identification of polynomial Wiener systems. *IEEE Trans. Signal Processing*, **49**(2), 301–313.

Chiang, H., C.L. Nikias, A.N. Venetsanopoulos (1986). Efficient implementation of quadratic digital filters. *IEEE Trans. Acoust., Speech, Signal Processing*, **34**(6), 1511–1528.

Dokic, M.V., P.M. Clarkson (1993). On the performance of a second-order adaptive Volterra filter. *IEEE Trans. Signal Processing*, **41**(2), 1944–1947.

Frank, W. (1994). MMD-A_n efficient approximation to the second order Volterra filter. In *Proc. ICASSP*, Vol. 3, pp. 517–520.

Korenberg, M.J. (1991). Orthogonal approaches to time-series analysis and system identification. *IEEE Signal Processing Mag.*, **8**(1), 29–43.

Ljung, L. (1987). *System Identification Theory for the User*. Englewood Cliffs, N.J, Prentice-Hall.

Mathews, V.J., J. Lee (1988). A fast recursive least squares second order Volterra filter. In *Proc. of ICASSP*, pp. 1383–1386.

Pitas, I., A.N. Venetsanopoulos (1990). *Nonlinear Digital Filters: Principles and Applications.* Kluwer Academic.

Panicker, T.M., V.J. Mathews (1996). Parallel-cascade realizations and approximations of truncated Volterra systems. In *Proc. ICASSP*, pp. 412–415.

Perry, F.A., Parker (1980). Adaptive solution of multichannel lattice models for linear and nonlinear systems. In *Proceedings of ISCAS*, pp. 744–746.

Priestley, M.B. (1998). *Non-linear and Non-stationary Time Series Analysis*. Academic Press Inc.

Priestley, M.B. (1998). State dependent models: a general approach to nonlinear time series analysis. *Journal of Time Series Analysis*, **1**(1), 47–71.

Raz, G.M., B.D. Van Veen (1998). Baseband Volterra filters for implementing carrier based nonlinearities. *IEEE Trans. Signal Processing*, **6**(1), 103–114.

Schetzen, M. (1980). *The Volterra and Wiener Theories of Nonlinear Systems*. Malabar, FL, Krieger.

K. Kazlauskas received Ph.D. degree from Kaunas Polytechnic Institute (Kaunas, Lithuania) in 1975. He is a senior researcher of the Process Recognition Department at the Institute of Mathematics and Informatics and Associate Professor at the Vilnius Pedagogical University. His research interests include design of concurrent algorithm and architectures for signal processing, and computer aided design of signal processing systems.

J. Kazlauskas received MSc degree from Vilnius University in 2000. Now he is an assistant of the Process Recognition Department at the Institute of Mathematics and Informatics. His scientific interests include digital signal processing and design of signal processing systems.

Sluoksniuotų polinominių filtrų struktūros

Kazys KAZLAUSKAS, Jaunius KAZLAUSKAS

Parodyta, kad netiesiniai Voltero, polinominiai autoregresijos ir bitiesiniai filtrai turi tokią pačią sluoksniuotą realizavimo procedūrą. Netiesiškumo eilė padidinama į filtro struktūrą pridedant daugiau sluoksnių. Struktūra yra modulinė ir sudaryta iš paprastų slenkamojo vidurkio arba autoregresijos filtrų. Modulinės sluoksniuotos struktūros gerai tinka realizuojant polinominius netiesinius filtrus labai didelės integracijos schemose.