BORDER EFFECT REMOVAL FOR IFIR AND INTERPOLATED VOLTERRA FILTERS

Eduardo L. O. Batista, Orlando J. Tobias, and Rui Seara

LINSE – Circuits and Signal Processing Laboratory
Department of Electrical Engineering
Federal University of Santa Catarina
88040-900 – Florianópolis - SC – Brazil
E-mails: {dudu, orlando, seara}@linse.ufsc.br

ABSTRACT

This paper presents a procedure to remove the border effect from interpolated finite impulse response (IFIR) and interpolated Volterra adaptive filters using the LMS algorithm. The used approach permits to reduce the steady-state mean-square error (MSE) of such structures. In situations that the border effect is important, the obtained improvement is noticeable. In addition, the computational burden required to implement such a procedure is slight. Simulation results confirm the effectiveness of the proposed approach.

Index Terms—Adaptive filters, adaptive signal processing, interpolation, least-mean-square methods, nonlinear filters.

1. INTRODUCTION

The interpolated finite impulse response (IFIR) filter is an approach aiming to implement effectively FIR filters [1]. Starting with the work of Neuvo *et al.* [1], much research effort has been carried out about IFIR filters, being successfully used in a variety of applications, such as line echo canceling [2], active control [3], among others. Such a filter is implemented by cascading a sparse FIR filter with an interpolator filter. The former reduces the number of coefficients while the latter recreates the removed coefficients by interpolation. The adaptive version of an IFIR (AIFIR) filter represents an alternative solution to the conventional adaptive FIR filters, particularly for applications in which a large number of taps is required [2]. In nonlinear applications, which are coefficient demanding, the need for reducing the computational burden points to the use of Volterra filters [4], [5]. In this case, to consider sparsity and interpolation is even more interesting because of the exponential coefficient reduction obtained by using interpolated structures [4], [5]. However, a drawback of such structures is its higher steady-state mean-square error (MSE) as compared with that of the standard implementation. This higher MSE has two main causes: the sparsity effect of the structure itself and the imperfections resulting from the equivalent filter composition, which is obtained from the convolution between the sparse and interpolator filters. The former is inherent to the interpolated approach and there is no means to reduce its impact. The latter is due to the convolution of two FIR filters of dimension M and N, resulting in an $(M+N-1)$-dimensional filter. In this way, the equivalent interpolated structure presents a border effect and a larger memory size [5]. Specifically, the border effect is the second source of performance loss. Thus, this work presents a procedure to implement interpolated adaptive filters (using the LMS algorithm) mitigating such an effect, which results in a significant performance improvement at the expense of a slight computational complexity overhead.

This paper is organized as follows. Section 2 introduces the general concept and mathematical description of interpolated FIR filters. Section 3 presents the procedure used for removing the border effect in IFIR structures. Section 4 extends such a procedure to interpolated Volterra filters. Section 5 shows some simulation results ratifying the good performance of the proposed approach. Finally, Section 6 concludes this paper.

This work was supported in part by the Brazilian National Research Council for Scientific and Technological Development (CNPq) and Studies and Projects Funding Body (FINEP).

2. IFIR FILTERS

Fig. 1 shows the block diagram of an IFIR filter. In this figure, $\mathbf{w}_s$ represents the coefficient vector characterizing the sparse filter and $\mathbf{i} = [i_0\, i_1 \cdots i_{M-1}]^T$ denotes an M-dimensional vector representing the interpolator filter. The input signal $x(n)$ and its interpolated version $\tilde{x}(n)$ are related by

$$\tilde{x}(n) = x(n) * \mathbf{i} = \sum_{j=0}^{M-1} i_j x(n-j)\,. \tag{1}$$

The sparse filter output is given by

$$\hat{y}(n) = \tilde{x}(n) * \mathbf{w}_s \tag{2}$$

where $*$ denotes the convolution operator.

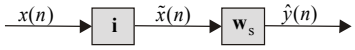

Fig. 1. Block diagram of an IFIR filter.

The factor determining the IFIR filter sparsity is termed interpolation factor, denoted by L [6]. The sparse filter vector $\mathbf{w}_s$ is obtained by setting to zero $(L-1)$ samples from each L consecutive ones from the original full vector $\mathbf{w} = [w(0)\, w(1) \cdots w(N-1)]^T$, with N representing the memory size of the original filter. Thus, we have

$$\mathbf{w}_s = \{w(0)\ 0\ \cdots\ w(L)\ 0\ \cdots\ w[(N_s-1)L]\ 0\ \cdots\ 0\}^T \tag{3}$$

with the corresponding input vector given by

$$\tilde{\mathbf{x}}(n) = [\tilde{x}(n)\ \ \tilde{x}(n-1)\ \ \tilde{x}(n-2)\ \ \cdots\ \ \tilde{x}(n-N+1)]^T\,. \tag{4}$$

In (3), N_s denotes the number of nonzero coefficients of the sparse filter, expressed as

$$N_s = \left\lfloor \frac{N-1}{L} \right\rfloor + 1 \tag{5}$$

where $\lfloor \cdot \rfloor$ represents the truncation operation. Then, considering (3) and (4), one can rewrite (2) as

$$\hat{y}(n) = \tilde{\mathbf{x}}^{\mathrm{T}}(n)\mathbf{w}_{\mathrm{s}} \tag{6}$$

which is the sparse filter output in a vector form. Note that in (1) and (6) the convolution operation is present. Now, defining an $(N+M-1)\times N$ interpolation matrix as

$$\mathbf{I} = \begin{bmatrix} i_0 & 0 & 0 & \cdots & 0 \\ i_1 & i_0 & 0 & \cdots & 0 \\ i_2 & i_1 & i_0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ i_{M-1} & i_{M-2} & i_{M-3} & \cdots & i_0 \\ 0 & i_{M-1} & i_{M-2} & \cdots & i_1 \\ 0 & 0 & i_{M-1} & \cdots & i_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & i_{M-1} \end{bmatrix} \tag{7}$$

and an $(N+M-1)\times 1$ extended input vector

$$\mathbf{x}_{\mathrm{e}}(n) = \begin{bmatrix} x(n) & x(n-1) & \cdots & x(n-N-M+2) \end{bmatrix}^{\mathrm{T}} \tag{8}$$

we can rewrite (4) as

$$\tilde{\mathbf{x}}(n) = \mathbf{I}^{\mathrm{T}}\mathbf{x}_{\mathrm{e}}(n)\ . \tag{9}$$

Thus, based on these definitions the output of the IFIR filter, written as a matrix product, is given by

$$\hat{y}(n) = \mathbf{x}_{\mathrm{e}}^{\mathrm{T}}(n)\mathbf{I}\mathbf{w}_{\mathrm{s}} = \mathbf{x}_{\mathrm{e}}^{\mathrm{T}}(n)\mathbf{w}_{\mathrm{i}}\ . \tag{10}$$

Note from (10) that the output signal $\hat{y}(n)$ is obtained from the extended input vector $\mathbf{x}_{\mathrm{e}}(n)$ and the coefficient vector $\mathbf{w}_{\mathrm{i}} = \mathbf{I}\mathbf{w}_{\mathrm{s}}$. For instance, if we consider a 3×1 sparse filter with $L=2$ and $\mathbf{i} = [0.5 \ \ 1 \ \ 0.5]^{\mathrm{T}}$ (linear interpolator [1], [6]), the sparse coefficient vector is given by

$$\mathbf{w}_{\mathrm{s}} = \begin{bmatrix} w(0) & \boxed{0} & w(2) \end{bmatrix}^{\mathrm{T}} \tag{11}$$

and the equivalent coefficient vector is

$$\mathbf{w}_{\mathrm{i}} = \begin{bmatrix} \underline{0.5w(0)} & w(0) & \boxed{0.5w(0)+0.5w(2)} & w(2) & \underline{0.5w(2)} \end{bmatrix}^{\mathrm{T}}. \tag{12}$$

Note from (12) that the center coefficient (boxed), coming from (11), is recreated as the arithmetic mean of its two neighbors. The underlined coefficients in (12) come from the convolution border effect. Notice also the larger length of the equivalent filter.

3. IFIR FILTERS WITH REMOVED BORDER EFFECT

The border effect indicated in (12) often produces a higher MSE in IFIR adaptive filters. Consider, for instance, the use of an IFIR filter with $N=3$, $L=2$ and $\mathbf{i} = [0.5 \ \ 1 \ \ 0.5]^{\mathrm{T}}$ for modeling a FIR filter with an exponential decreasing impulse response, typical in several practical applications. In this case, the first coefficient in (12) may have a significant value, originating a considerable modeling error. To solve this problem, the equivalent coefficient vector is changed to

$$\mathbf{w}'_{\mathrm{i}} = \begin{bmatrix} w(0) & \boxed{0.5w(0)+0.5w(2)} & w(2) \end{bmatrix}^{\mathrm{T}} \tag{13}$$

or equivalently

$$\mathbf{w}'_{\mathrm{i}} = \begin{bmatrix} i_1 w(0) & i_0 w(0) + i_2 w(2) & i_1 w(2) \end{bmatrix}^{\mathrm{T}} \tag{14}$$

for a generalized interpolator with $M=3$ and coefficients $\mathbf{i} = [i_0 \ \ i_1 \ \ i_2]^{\mathrm{T}}$. To obtain (14), we start by determining a transformation matrix satisfying the following relation:

$$\mathbf{w}'_{\mathrm{i}} = \mathbf{T}\mathbf{w}_{\mathrm{i}} \tag{15}$$

with $\mathbf{T}$ denoting the transformation matrix. By considering the dimensions of the involved vectors [$N\times 1$ for $\mathbf{w}'_{\mathrm{i}}$ and $(N+M-1)\times 1$ for $\mathbf{w}_{\mathrm{i}}$] and the transformation nature (row permutation and elimination), the $N\times(N+M-1)$ transformation matrix has a similar structure to a permutation matrix [7]

$$\mathbf{T} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix}. \tag{16}$$

Now, substituting $\mathbf{w}_{\mathrm{i}} = \mathbf{I}\mathbf{w}_{\mathrm{s}}$ into (15), we obtain

$$\mathbf{w}'_{\mathrm{i}} = \mathbf{T}\mathbf{I}\mathbf{w}_{\mathrm{s}}\ . \tag{17}$$

Note, from $\mathbf{w}_{\mathrm{i}} = \mathbf{I}\mathbf{w}_{\mathrm{s}}$ and (17), that to obtain an IFIR filter with removed border effect (RBEIFIR), the interpolation matrix is replaced by

$$\mathbf{I}' = \mathbf{T}\mathbf{I}\ . \tag{18}$$

As a result, the input vector for the sparse filter in the RBEIFIR implementation is given by

$$\tilde{\mathbf{x}}'(n) = \mathbf{I}'^{\mathrm{T}}\mathbf{x}(n) = \mathbf{I}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\mathbf{x}(n) \tag{19}$$

with

$$\mathbf{x}(n) = [x(n) \ x(n-1) \ \cdots \ x(n-N+1)]^{\mathrm{T}}\ . \tag{20}$$

From (19) and considering that $\mathbf{i} = [i_0 \ \ i_1 \ \ i_2]^{\mathrm{T}}$, one obtains

$$\tilde{\mathbf{x}}'(n) = \begin{bmatrix} \tilde{x}(n) & \tilde{\mathbf{x}}_{\mathrm{c}}(n) & \bar{x}(n) \end{bmatrix}^{\mathrm{T}} \tag{21}$$

where

$$\tilde{\mathbf{x}}_{\mathrm{c}}(n) = \begin{bmatrix} \tilde{x}(n) & \tilde{x}(n-1) & \cdots & \tilde{x}(n-N+3) \end{bmatrix}^{\mathrm{T}} \tag{22}$$

$$\tilde{x}(n) = i_1 x(n) + i_2 x(n-1) \tag{23}$$

and

$$\bar{x}(n) = i_0 x(n-N+2) + i_1 x(n-N+1)\ . \tag{24}$$

Thus, to eliminate the border effect, we replace the sparse filter input vector (4) by the new one (21). By considering the required computational complexity, the computation of (23) and (24) implies on incrementing 4 multiplications and 2 additions per sample.

4. EXTENSION TO INTERPOLATED VOLTERRA FILTERS

A Volterra filter is a nonlinear structure formed by P filters in parallel. A linear or first-order and several nonlinear blocks with orders from 2 to P form the structure of a Volterra filter [8]. The interpolated Volterra filter is a reduced complexity implementation of a standard Volterra one, using sparsity to that end [4], [5]. The block diagram of such a structure is illustrated in Fig. 2, where $x(n)$ is the input signal, $\hat{y}(n)$ denotes the output signal, and $\mathbf{i}$ represents the interpolator filter. The sparse Volterra filter is denoted by $\mathbf{h}_{\mathrm{Vs}}$ and its block structure is pointed out by the dashed box, with each p^{th}-order sparse block denoted by $\mathbf{h}_{ps}$ with output given by $\hat{y}_p(n)$. Vectors $\tilde{\mathbf{x}}_p(n)$ are the p^{th}-order

interpolated input vectors. Since the first-order block of the interpolated Volterra filter is equivalent to the IFIR filter, the input vector for the first-order block $\tilde{\mathbf{x}}_1(n)$ is the same as (4). The remaining input vectors are obtained recursively by [5]

$$\tilde{\mathbf{x}}_p(n) = \tilde{\mathbf{x}}_1(n) \otimes \tilde{\mathbf{x}}_{p-1}(n). \tag{25}$$

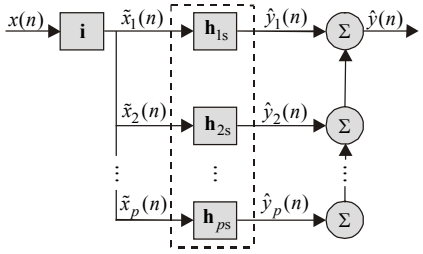

Fig. 2. Block diagram of the interpolated Volterra filter.

The first-order sparse coefficient vector $\mathbf{h}_{1s}$ has the same structure as given in (3) and the remaining ones are obtained as follows. The input-output relationship for each block of the interpolated Volterra filter is given by [5]

$$\hat{y}_p(n) = \mathbf{x}_{pe}^{\mathrm{T}}(n)\mathbf{I}_p\mathbf{h}_{ps} \tag{26}$$

with $\mathbf{I}_1$ obtained from (7) and

$$\mathbf{I}_p = \mathbf{I}_1 \otimes \mathbf{I}_{p-1}. \tag{27}$$

Vector $\mathbf{x}_{1e}(n)$ is determined from (8) and

$$\mathbf{x}_{pe}(n) = \mathbf{x}_{1e}(n) \otimes \mathbf{x}_{(p-1)e}(n). \tag{28}$$

The border effect removing procedure for the interpolated Volterra filter is the same as for IFIR filters just by changing the input vector to that given in (21). Thus, for the first-order block the results of Section 3 are used here. Considering however the second-order block, the borderless second-order input vector is obtained as

$$\tilde{\mathbf{x}}_2'(n) = \tilde{\mathbf{x}}'(n) \otimes \tilde{\mathbf{x}}'(n) = \mathbf{I}'^{\mathrm{T}}\mathbf{x}(n) \otimes \mathbf{I}'^{\mathrm{T}}\mathbf{x}(n) = \mathbf{I}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\mathbf{x}(n) \otimes \mathbf{I}^{\mathrm{T}}\mathbf{T}^{\mathrm{T}}\mathbf{x}(n). \tag{29}$$

By considering the Kronecker mixed-product rule [8], [5] in (29), we obtain

$$\tilde{\mathbf{x}}_2'(n) = [(\mathbf{I}^{\mathrm{T}} \otimes \mathbf{I}^{\mathrm{T}})(\mathbf{T}^{\mathrm{T}} \otimes \mathbf{T}^{\mathrm{T}})][\mathbf{x}(n) \otimes \mathbf{x}(n)] \tag{30}$$

resulting in

$$\tilde{\mathbf{x}}_2'(n) = \mathbf{I}_2^{\mathrm{T}}\mathbf{T}_2^{\mathrm{T}}\mathbf{x}_2(n) \tag{31}$$

with $\mathbf{T}_2 = \mathbf{T} \otimes \mathbf{T}$, $\mathbf{I}_2 = \mathbf{I} \otimes \mathbf{I}$, and $\mathbf{x}_2(n) = \mathbf{x}(n) \otimes \mathbf{x}(n)$ as the second-order Volterra input vector. By using (31), the input-output relationship for the borderless second-order block is given by

$$\hat{y}_2'(n) = \mathbf{x}_2^{\mathrm{T}}(n)\mathbf{T}_2\mathbf{I}_2\mathbf{h}_{2s} \tag{32}$$

resulting in the equivalent coefficient vector

$$\mathbf{h}_{2i}' = \mathbf{T}_2\mathbf{I}_2\mathbf{h}_{2s}. \tag{33}$$

By evaluating (33) for the case $L = 2$ and $N = 3$, using the linear interpolator $\mathbf{i} = [0.5 \;\; 1 \;\; 0.5]^{\mathrm{T}}$, and organizing the resulting vector in a matrix form [5], we get

$$\mathbf{H}_{2i}' = \begin{bmatrix} h(0,0) & 0.5h(0,0)+0.5h(0,2) & h(0,2) \\ \begin{matrix}0.5h(0,0)+\\ 0.5h(2,0)\end{matrix} & \begin{matrix}0.25h(0,0)+0.25h(0,2)+\\ 0.25h(2,2)+0.25h(2,0)\end{matrix} & \begin{matrix}0.5h(0,2)+\\ 0.5h(2,2)\end{matrix} \\ h(2,0) & 0.5h(2,0)+0.5h(2,2) & h(2,2) \end{bmatrix}. \tag{34}$$

Expression (34) is the equivalent second-order matrix with no border effect and equivalent reduced memory size of $N \times N$. The extension to higher order blocks is straightforward, resulting in the following generalized equivalent coefficient vector:

$$\mathbf{h}_{pi}' = \mathbf{T}_p\mathbf{I}_p\mathbf{h}_{ps} \tag{35}$$

with $\mathbf{T}_p = \mathbf{T} \otimes \mathbf{T}_{p-1}$ and $\mathbf{I}_p = \mathbf{I} \otimes \mathbf{I}_{p-1}$.

5. SIMULATION RESULTS

In this section, some simulation results are presented in order to illustrate the proposed procedure. The examples compare results from a system identification problem [9], using interpolated Volterra filters with and without border effect removal. For all examples a white Gaussian input signal with variance $\sigma_x^2 = 1$, a linear interpolator $\mathbf{i} = [0.5 \;\; 1 \;\; 0.5]^{\mathrm{T}}$, and $L = 2$ are used.

Example 1: In this example, the plant to be modeled is a decaying exponential function for the first-order block and a decaying exponential surface for the second-order one. The memory size is $N = 11$ and both filters are adapted by using the LMS algorithm with $\mu = 0.2\mu_{\max}$, where $\mu_{\max}$ is the maximum value of the step-size parameter for algorithm convergence (experimentally obtained). The MSE results are shown in Fig. 3. Note from that figure that the approach without border effect has a lower steady-state MSE. Fig. 4 shows the first-order coefficients of the plant and the steady-state equivalent ones for each structure. Figs. 5 and 6 illustrate the second-order kernel shown as a surface plot. Note that the matching between plant and estimated kernels is better when the border effect is removed. In addition, a lower memory size is required for the proposed approach.

Example 2: In this example, a second-order plant having larger coefficient values in the center of the kernel is considered. In this case, the coefficients arising from border effects are small. The MSE curve is shown in Fig. 7. Note that in this case the performance improvement is smaller than that obtained in Example 1. This is due to the plant characteristics. The first-order block curves are shown in Fig. 8 and second-order kernel surfaces, in Figs. 9 and 10. In that case, because of the used plant characteristics, the differences between model and estimated kernels are less visible; however, noticeable differences in the MSE curves are still verified.

6. CONCLUSIONS

In this work, a procedure to improve the MSE performance of IFIR and interpolated Volterra adaptive filters has been discussed. Such a procedure requires a slight increment on the computational burden while producing a significant improvement on the MSE performance of the interpolated filters. Simulation results attest the effectiveness of the proposed approach.

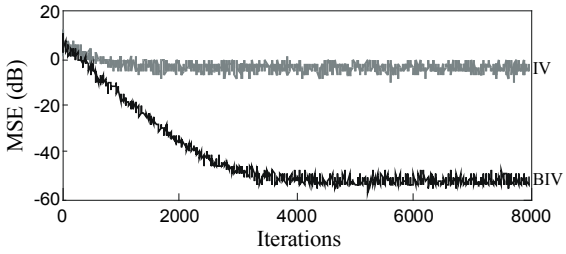

Fig. 3. Example 1. MSE curves (200 independent runs). (IV) Interpolated Volterra filter. (BIV) Borderless interpolated Volterra filter.

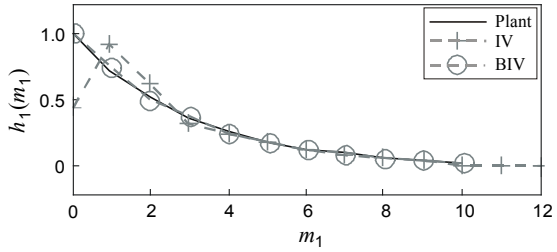

Fig. 4. Coefficient curves for the first-order block from Example 1.

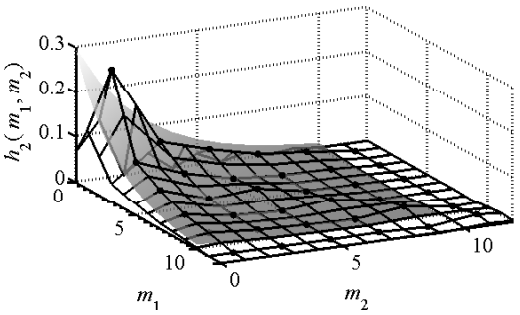

Fig. 5. Superposition of second-order coefficient surfaces for Example 1. (Solid surface) plant; (wireframe) interpolated Volterra filter with main coefficients as the vertices indicated by black dots and the remaining ones are recreated.

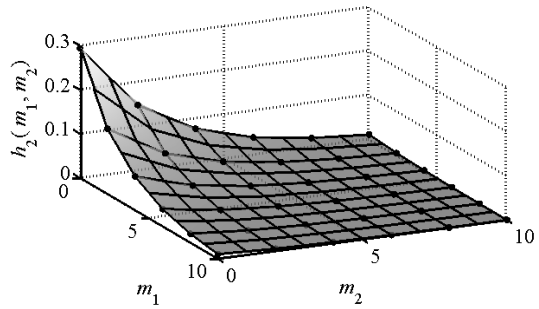

Fig. 6. Superposition of second-order coefficient surfaces for Example 1. (Solid surface) plant; (wireframe) borderless interpolated Volterra filter with main coefficients as the vertices indicated by black dots and the remaining ones are recreated.

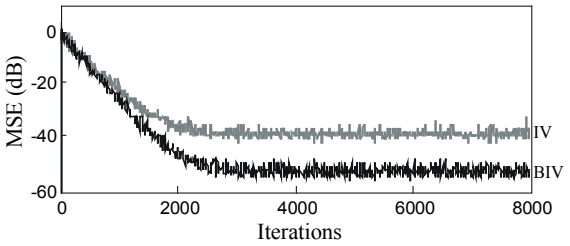

Fig. 7. MSE curves for Example 2 (200 independent runs). (IV) Interpolated Volterra filter. (BIV) Borderless interpolated Volterra filter.

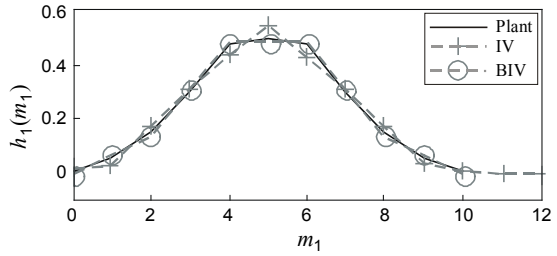

Fig. 8. Coefficient curves for the first-order block from Example 2.

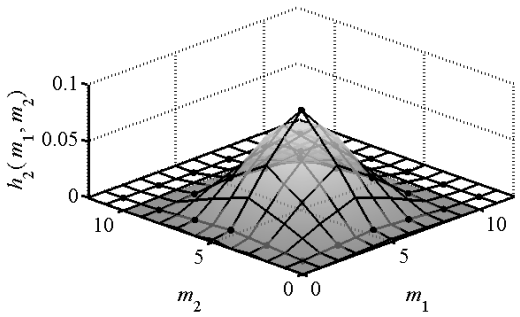

Fig. 9. Superposition of second-order coefficient surfaces for Example 2. (Solid surface) plant; (wireframe) interpolated Volterra filter with main coefficients as the vertices indicated by black dots and the remaining ones are recreated.

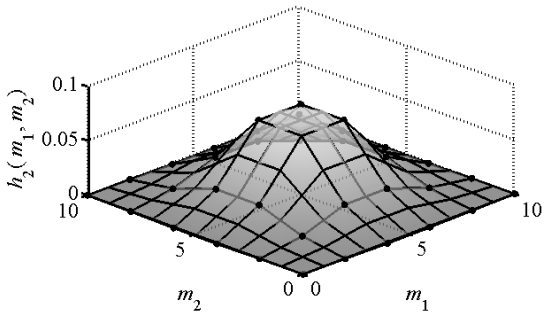

Fig. 10. Superposition of second-order coefficient surfaces for Example 2. (Solid surface) plant; (wireframe) borderless interpolated Volterra filter with main coefficients as the vertices indicated by black dots and the remaining ones are recreated.

7. REFERENCES

[1] Y. Neuvo, C. Y. Dong, and S. K. Mitra, "Interpolated finite impulse response digital filters," *IEEE Trans. Acoust., Speech, Signal Process*, vol. 32, no. 3, pp. 563-570, June 1984.

[2] A. Abousaada, T. Aboulnasr, and W. Steenaart, "An echo tail canceller based on adaptive interpolated FIR filtering," *IEEE Trans. Circ. Syst. II*, vol. 39, no. 7, pp. 409-416, July 1992.

[3] S. Kuo and D. R. Morgan, *Active Noise Control Systems*. John Wiley & Sons, 1996.

[4] E. L. O. Batista, O. J. Tobias, and R. Seara, "Fully and partially interpolated adaptive Volterra filters," in *Proc. European Signal Processing Conf.*, Antalya, Turkey, CD-ROM, paper no. 1507, Sept. 2005, pp. 1-4

[5] E. L. O. Batista, O. J. Tobias, and R. Seara, "A mathematical framework to describe interpolated adaptive Volterra filters," in *Proc. IEEE Int. Telecomm. Symp.*, Fortaleza, Brazil, Sept. 2006, pp. 144-149.

[6] O. J. Tobias and R. Seara, "Analytical model for the first and second moments of an adaptive interpolated FIR filter using the constrained filtered-X LMS algorithm," *IEE Proceedings – Vision, Image, Signal Process.*, vol. 148, no. 5, pp. 337-347, Oct. 2001.

[7] T. K. Moon and W. C. Stirling, *Mathematical Methods and Algorithms for Signal Processing*, Prentice-Hall, 2000.

[8] V. J. Mathews and G. L. Sicuranza, *Polynomial Signal Processing*, John Wiley & Sons Inc., 2000.

[9] S. Haykin, *Adaptive Filter Theory*, 4th ed., Prentice-Hall, 2002.