PAPER Special Section of Papers Selected from ITC-CSCC'96

2-D Pipelined Adaptive Filters Based on 2-D Delayed LMS Algorithm

Katsushige MATSUBARA†, *Student Member*, Kiyoshi NISHIKAWA†, *and* Hitoshi KIYA†, *Members*

SUMMARY A pipelined adaptive digital filter (ADF) architecture based on a two-dimensional least mean square algorithm is proposed. This architecture enables the ADF to be operated at a high clock rate and reduction of the required amount of hardware. To achieve this reduction we introduce a new building unit, called a block, and propose implementing the pipelined ADF using the block. Since the number of blocks in a cell is adjustable, we derive a condition for satisfying given specifications. We show the smallest number of blocks and the corresponding delay can be determined by using the proposed method.

key words: *pipelining, two-dimensional LMS algorithm, delayed LMS algorithm*

1. Introduction

In this paper we propose a two-dimensional (2-D) pipelined adaptive digital filter (ADF) architecture based on 2-D least mean square (LMS) algorithm [1].

2-D ADFs are used for image processing, such as enhancement and data compression [2], [3]. Processing of 2-D signals requires a much larger amount of calculations than one-dimensional (1-D) signals. In addition, in most cases, 2-D signals are required to be processed at a higher clock rate than 1-D signals. A 2-D ADF is therefore required to have the capability of processing a large amount of calculations per clock period to maintain high throughput. An architecture which can provide high throughput and a small amount of hardware is therefore required. For satisfying this requirement, dedicated LSI of an ADF is considered to be one effective method to satisfy this requirement [4]. There are, however, no proposals of 2-D ADF architectures so far, although it is known that the hardware implementations of 1-D ADFs [7]–[9] are effective for high-clock-rate 1-D applications.

In this paper we propose a 2-D pipelined ADF architecture, which is an extended version of the 1-D pipelined ADF architecture [9] based on the delayed LMS (DLMS) algorithm [5]. The proposed architecture enables an ADF to provide high throughput. By applying the proposed architecture we can reduce the required amount of hardware. The reduction is achieved by introducing a new processing module which we call a *block*. A block consists of several taps of an ADF. By cascading the blocks, a pipelining element of the 2-D ADF, a cell, is constructed.

The proposed architecture has the capability to adjust the number of blocks in a cell and to satisfy the above two requirements by selecting the number of blocks properly. We derive the condition for obtaining the smallest number of blocks which satisfies the required throughput. We show that the amount of hardware can be reduced by using the proposed architecture.

In Sect. 2, a review of the 1-D pipelined ADF based on the DLMS algorithm as a preliminary is given. In Sect. 3, the pipelined 2-D LMS algorithm is proposed. In Sect. 4, a 2-D pipelined ADF architecture based on the proposed algorithm is considered. In Sect. 5, an efficient technique for reducing the amount of hardware is provided. Conclusions are given in Sect. 6.

2. 1-D Pipelined ADF

Here we briefly describe the DLMS algorithm and its pipeline implementation in a 1-D case.

2.1 Delayed LMS Algorithm

The DLMS algorithm [5] has the delay D in the error feedback path, as shown in Fig. 1. The DLMS for an N-tap adaptive FIR filter can be described by

$$w_k(i+1) = w_k(i) + \mu e(i-D)x(i-k-D) \quad \{0 \leqq k \leqq N-1\} \tag{1}$$

$$e(i-D) = d(i-D) - y(i-D) \tag{2}$$

$$y(i-D) = \sum_{k=0}^{N-1} w_k(i-D)x(i-k-D) \tag{3}$$

where

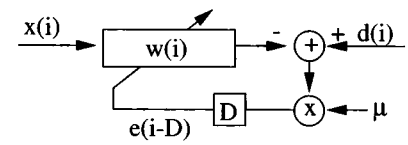


Fig. 1 Block diagram of the DLMS algorithm.

Manuscript received October 4, 1996.
Manuscript revised January 7, 1997.

†The authors are with the Graduate School of Engineering, Tokyo Metropolitan University, Hachioji-shi, 192–03 Japan.

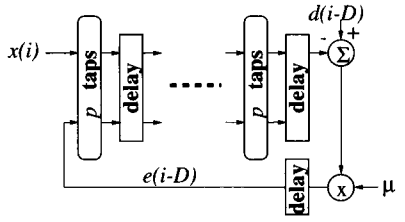


Fig. 2 Block diagram of 1-D pipelined ADF using grouping technique.

i : time index
$w_k(i)$: k-th coefficient of the filter
μ : the stepsize parameter of the algorithm
$e(i)$: the error signal defined by Eq. (2)
$d(i)$: the desired signal at time i
$x(i)$: the input signal at time i
$y(i)$: the filter output signal.

The delay D enables an ADF to be operated in pipelined processing, and there are several proposals for the architecture of a 1-D pipelined ADF. It is known, however, that the convergence of the DLMS becomes poor as D increases because the upper boundary of μ becomes lower as D increases to guarantee the convergence of the mean-squared error (MSE) of the DLMS [5], [6].

2.2 Pipelined Architecture

Pipelining of the DLMS is accomplished by moving delays and placing them at every tap of the filter [7], [8]. When applying this method, the delay is determined by

$$D = N + 1. \tag{4}$$

An architecture using this method achieves very high throughput. The delay becomes larger, however, as the length of the ADF increases. One problem is that the convergence property becomes poor if the length of the ADF is large.

To solve this problem we proposed a new architecture [9]. In this architecture, the delays are placed at every p tap instead of every tap. The upper boundary of μ becomes higher as D decreases. Thus, the convergence property can be improved as D decreases. Figure 2 shows the block diagram of this architecture. The delay of this architecture is determined by

$$D = \left\lceil \frac{N}{p} \right\rceil + 1 \tag{5}$$

where $\lceil x \rceil$ is the smallest integer greater than x. Using scheduling and reusing techniques, we can also reduce the required amount of hardware with maintaining high throughput [9]. Note that when $p = 1$, this architecture reduces to the architecture proposed in [7], [8].

In following sections we will derive a 2-D pipelined ADF which is an extended version of the 1-D pipelined ADF described in [9].

3. Pipelined 2-D LMS Algorithm

Here we describe the 2-D LMS algorithm which is used in the proposed pipelined ADF.

3.1 2-D LMS Algorithm

In 1-D signal processing the LMS algorithm has only one direction for its adaptation. In 2-D signal processing there is some freedom to select a direction for adaptation. There are therefore several proposals of the 2-D LMS algorithm [1], [10]. In this paper we use the algorithm described in [1] because it has a simple form and is therefore suitable for hardware implementation.

The algorithm is described by the following Eq. [1].

$$w_{k,l}(j+1) = w_{k,l}(j) + \mu e(j) x(m-k, n-l) \quad \{0 \leqq k \leqq N-1, 0 \leqq l \leqq N-1\} \tag{6}$$

$$j = mM + n \tag{7}$$

$$e(j) = d(m,n) - y(m,n) \tag{8}$$

$$y(m,n) = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} w_{k,l}(j) \cdot x(m-k, n-l) \tag{9}$$

where

j : iteration number
N : the filter size
M : the size of input plane
$w_{k,l}(j)$: filter coefficients at the j-th iteration
μ : the step size of filter adaptation
$x(m,n)$: the input signal
$d(m,n)$: the desired signal
$e(j)$: the error signal defined by Eq. (8)
m : index of vertical direction
n : index of horizontal direction.

Because there is some freedom to select a direction of the processing of filtering, we assume that the filter processes the signal along n-direction for simplification of pipeline implementation. Under this assumption we can rewrite Eq. (9) as

$$y(m,n) = \sum_{k=0}^{N-1} y_k(m,n) \tag{10}$$

$$y_k(m,n) = \sum_{l=0}^{N-1} w_{k,l}(j) x(m-k, n-l). \tag{11}$$

Equation (11) has a similar form as that of a 1-D convolution sum. This enables us to use algorithms that are used for pipelining 1-D ADFs to construct 2-D pipelined ADFs.

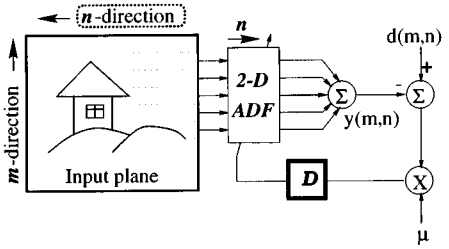


Fig. 3 Block diagram of 2-D LMS with delayed adaptation.

3.2 2-D LMS with Delayed Adaptation

For pipelining Eq. (6), let us consider using the idea of the DLMS algorithm [5]. By applying the DLMS to weight update Eq. (6), we get the following equation:

$$w_{k,l}(j+1) = w_{k,l}(j) + \mu e(j-D)x(\tilde{m}-k, \tilde{n}-l) \quad \{0 \leqq k \leqq N-1, 0 \leqq l \leqq N-1\} \quad (12)$$

$$\tilde{m} = \left\lfloor \frac{j-D}{M} \right\rfloor \quad (13)$$

$$\tilde{n} = \mathrm{rem}\,(j-D, M) \quad (14)$$

where $\lfloor x \rfloor$ is the largest integer less than x, and $\mathrm{rem}(x, y)$ is a remainder after x divided by y. D is the delay in weight adaptation and is inserted in the error feedback path along n-direction. The delay D enables an ADF to be operated in pipelined processing as in the 1-D case. The convergence of algorithms becomes poor, however, as D increases. Figure 3 shows a block diagram of the 2-D LMS based on Eq. (12).

In the next section we describe the proposed 2-D pipelined ADF architecture which realizes both a high throughput and a good convergence property.

4. The Proposed Architecture

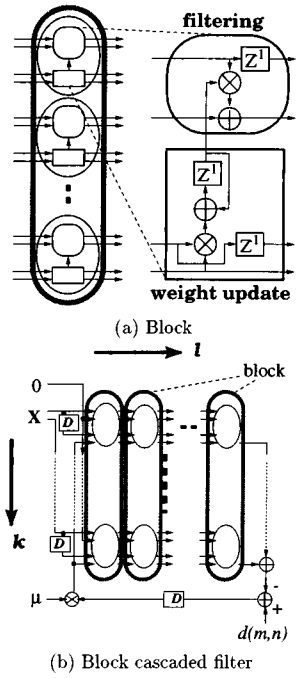
Here we describe the proposed 2-D pipelined ADF architecture. First, we develop a new building unit of the 2-D ADF which we call a block. Next, a 2-D pipelined ADF architecture is proposed. Then, we show computer simulation to verify the validity of the proposed architecture.

4.1 Block

Given Eq. (12), the proposed architecture is constructed. To extend the DLMS algorithm to 2-D we introduce a new processing module of the 2-D ADF which we call a block. Each block contains N taps of the ADF in m-direction. A block is shown in Fig. 4 (a). The l-th column of the coefficient matrix which corresponds to the l-th block, denoted by $\boldsymbol{b}_l(j)$, is described by

$$\boldsymbol{b}_l(j) = [w_{0,l}(j), w_{1,l}(j), \cdots, w_{N-1,l}(j)]^T \quad (15)$$

where $[\cdot]^T$ denotes transposition of a vector. Using the



(a) Block

(b) Block cascaded filter

Fig. 4 Data-flow graph of the block and its cascade form.

vector $\boldsymbol{b}_l(j)$, we can rewrite Eq. (12) as

$$\boldsymbol{b}_l(j+1) = \boldsymbol{b}_l(j) + \mu e(j-D)\boldsymbol{x}_l(j-D) \quad \{0 \leqq l \leqq N-1\} \quad (16)$$

where $\boldsymbol{x}_l(j-D)$ is defined as

$$\boldsymbol{x}_l(j-D) = [x(\tilde{m}, \tilde{n}-l), x(\tilde{m}-1, \tilde{n}-l), \cdots, x(\tilde{m}-N+1, \tilde{n}-l)]^T. \quad (17)$$

The form of Eq. (16) is similar to that of Eq. (1). As in the 1-D case, we can construct a 2-D ADF by cascading the blocks. Figure 4(b) shows an example of this structure.

4.2 The Proposed 2-D Pipelined ADF Architecture

Pipelining of the 2-D ADF is accomplished by moving delays around a data-flow graph and placing them at every p block [9]. A pipelining element, a cell, consists of the blocks between delays. Figure 5 shows block diagrams of a cell and the 2-D pipelined ADF. The finest grain of pipelining is achieved when each cell contains one block, that is $p=1$, while no pipelining is achieved if $p=N$. The pipelined ADF has the highest throughput if $p=1$.

4.3 Simulation Results

To verify the validity of the proposed architecture, we show results of computer simulations of system identification.

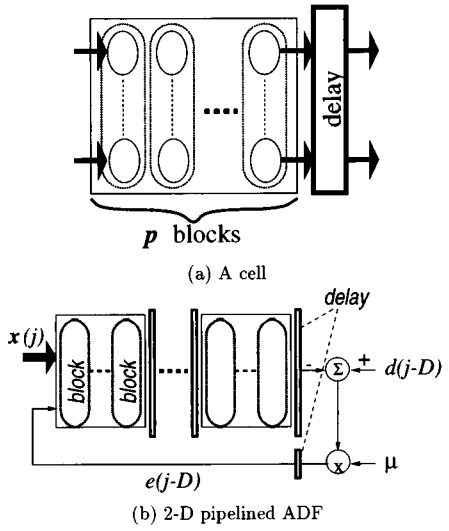


Fig. 5 Block diagrams of a pipelining unit, a cell, and 2-D pipelined ADF.

The optimum system, denoted by $\boldsymbol{W}_{opt}$, is a 2-D low-pass finite impulse response (FIR) filter of size 8-by-8. The order of the ADF $\boldsymbol{W}(j)$ is selected as the same size of $\boldsymbol{W}_{opt}$. The input signal is a 64-by-64 matrix whose elements are normally distributed with mean 0 and variance 1. A white Gaussian noise is added to the desired signal, and the signal to noise ratio is 80 dB. As the reference of comparison we use the impulse response error ratio (IRER) defined as

$$\mathrm{IRER}(j) = 10 \log_{10} \frac{\|\boldsymbol{W}_{opt} - \boldsymbol{W}(j)\|^2}{\|\boldsymbol{W}_{opt}\|^2} \tag{18}$$

where $\|\boldsymbol{X}\|^2 = \sum(\mathrm{diag}(\boldsymbol{X}^T\boldsymbol{X}))$ and $\mathrm{diag}[\cdot]$ is diagonals of a matrix $\boldsymbol{X}$.

Figure 6 shows learning curves for D =0, 8 and 16. The stepsize μ is selected to be the maximum value which guarantees the convergence of the MSE for the corresponding D [5]. From this figure we can see that the convergence property of the ADF is improved as D decreases because the upper boundary for the stepsize μ increases as D decreases. The proposed architecture, as described above, enables the ADF to have a small delay and therefore to get a better convergence property.

5. Efficient Structure

From previous sections we can see that a direct mapping implementation always requires N blocks. This means that the amount of hardware becomes larger as N, the length of the ADF, increases. In this section we show an effective technique of reusing the blocks for reducing the amount of hardware with keeping the required throughput. Then an example of hardware implementation is shown.

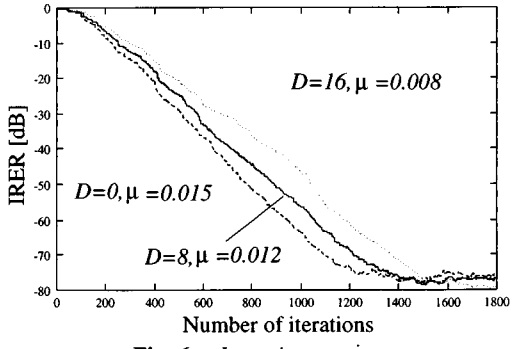


Fig. 6 Learning curves.

5.1 Preparation

For simplification of the discussion in this paper, we assume that the ADF treats only one frame of an image and that pipelined multipliers are used.

Let us consider specifications are given as C and C_k: C is the maximum allowable period for processing one frame and C_k is the clock period of an adder/multiplier in the block. From those specifications, the desired throughput is expressed as

$$T_P = \frac{M^2 C_k}{C}. \tag{19}$$

5.2 Scheduling of Blocks

We can derive the smallest amount of hardware which satisfies the desired throughput T_P in the following way. We define h_b as the number of blocks which can be used in each cell. Let us derive the condition for h_b to satisfy T_P. By extending the method of the 1-D pipelined ADF [9], we obtain the following condition

$$S_t \leq \frac{1}{T_P} \tag{20}$$

where

$$S_t = r + \alpha + \beta - 1 \tag{21}$$

$$r = \frac{\tau_m}{\tau_a} \tag{22}$$

$$\alpha = \left\lceil \frac{p}{h_b} \right\rceil \tag{23}$$

$$\beta = \lceil \log_2(p - (\alpha - 1)h_b + \lceil a_{\alpha-1} \rceil) \rceil \tag{24}$$

$$a_i = \frac{1 - h_b}{2^i} + h_b, \tag{25}$$

τ_m is the required period for processing one multiplication and τ_a is the required period for processing one addition.

Our purpose is to determine the smallest h_b satisfying the condition (20). Figure 7 shows an example of the characteristic of the number of blocks h_b versus the throughput $1/S_t$ and also shows the limit condition

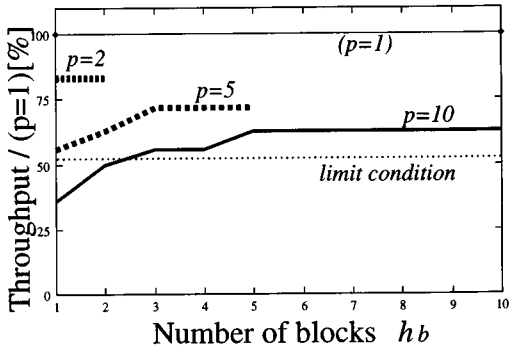


Fig. 7 Characteristic of h_b versus $1/S_t$.

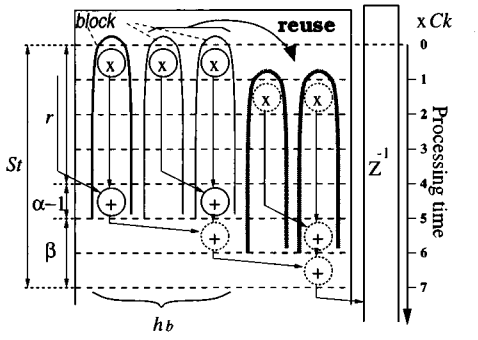


Fig. 8 Procedure of reusing the blocks under $p = 5$ and $h_b = 3$.

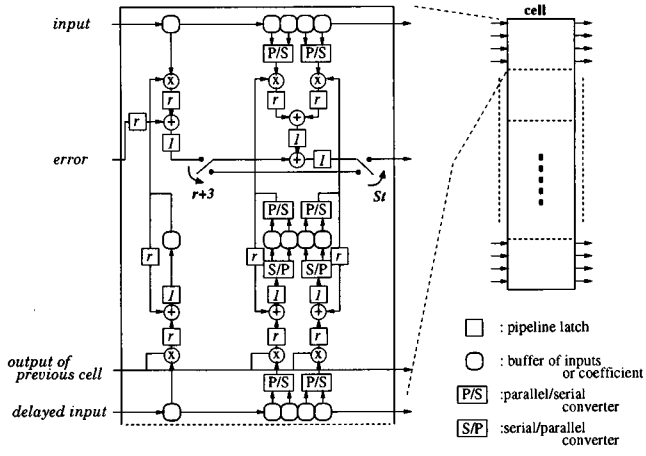


Fig. 9 Example of cell architecture under $p = 5$ and $h_b = 3$.

determined from the condition (20). A detailed description of Fig. 7 is given in the next subsection. When specifications are given, we obtain the limit condition and the characteristic like Fig. 7, and from those figures we can select the smallest h_b on each p. Note that when h_b is less than p, a block is reused. Figure 8 shows an example of the procedure of reusing blocks. In Fig. 8, two blocks are reused and their multipliers and adders are scheduled. An example of the proposed architecture is shown in Fig. 9. The delay in Eq. (12) is automatically determined by

$$D = \left\lceil \frac{N}{p} \right\rceil + \left\lceil \frac{N}{S_t} \right\rceil. \tag{26}$$

If m and h_b are selected properties, the pipelined ADF is able to have the smallest D with maintaining the desired throughput.

According to the above procedure we can determine h_b and, hence, D. The pipelined ADF can be implemented using those values which require the minimum hardware under the given specifications.

5.3 Design Example

Let us consider implementing the pipelined ADF under the conditions $M = 512$, $C = 1/60$ sec, $N = 30$, $r = 4$, and $C_k = 6.7$ nsec. C_k is determined by the result of a design using the PARTHENON [11]. In Fig. 7, the dotted line shows limited throughput under those conditions. From this figure we know that the required block per cell is $h_b = 3$ if $p = 10$, $h_b = 1$ if $p = 5$ or $p = 2$, while direct mapping form requires $h_b = p$. The delay D is reduced to 7, 10 and 20, respectively. By applying the proposed architecture, the pipelined ADF can have a small amount of hardware and a good convergence property with satisfying the desired throughput.

6. Conclusion

We proposed a pipelined ADF architecture based on a 2-D LMS algorithm. We showed that the proposed architecture enables an ADF to have both a high throughput and a small amount of hardware. To achieve the hardware reduction we proposed to implement the pipelined ADF using a block as the constructing unit. The ADF is pipelined by cascading these blocks. We showed the number of blocks in a cell is adjustable so that we could derive the condition for satisfying given specifications. We showed the smallest number of blocks and the corresponding delay can be determined by using the proposed architecture. An example of the hardware design was shown.

Acknowledgement

The authors would like to express their thanks to Software Research Associates, Inc., NTT Communication Science Laboratory.

References

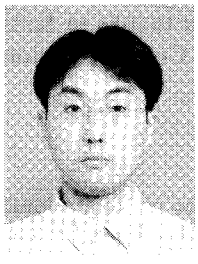
[1] M.M. Hadhoud and D.W. Thomas, "The two-dimensional adaptive LMS (TDLMS) algorithm," IEEE Trans. Circuits & Syst., vol.35, no.5, pp.485–494, May 1988.

[2] H. Youlal, M. Janati-I, and M. Najim, "Two-dimensional joint process lattice for adaptive restoration of images," IEEE Trans. Image Proce., vol.1, no.3, pp.366–378, July 1992.

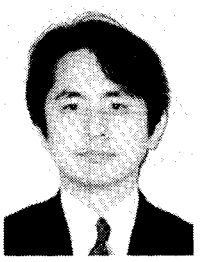
[3] P.A. Maragos, R.W. Schafer, and R.M. Mersereau, "Two-

dimensional linear prediction and its application to adaptive predictive coding of images," IEEE Trans. Acoust., Speech & Signal Process., vol.32, no.6, pp.1213–1229, Dec. 1984.
[4] Y. Iwata, M. Kawamata, and T. Higuchi, "Block parallel architecture of separable denominator 2-D state-space digital filters based on the reduced-dimensional decomposition," Trans. of IEICE, vol.J77-A, no.4, pp.635–643, April 1994.
[5] G. Long, F. Ling, and J.G. Proakis, "The LMS algorithm with delayed coefficient adaptation," IEEE Trans. Acoust., Speech & Signal Process., vol.37, no.9, pp.1397–1405, Sept. 1989.
[6] G. Long, F. Ling, and J.G. Proakis, "Corrections to "The LMS algorithm with delayed coefficient adaptation"," IEEE Trans. Signal Process., vol.40, no.1, pp.230–232, Jan. 1992.
[7] H. herzberg, R. Haimi-Cohen, and Y. Be'ery, "A systolic array realization of an LMS adaptive filter and the effects of delayed adaptation," IEEE Trans. Signal Process., vol.40, no.11, pp.2799–2803, Nov. 1992.
[8] M.D. Meyer and D.P. Agrawal, "A high sampling rate delayed LMS filter architecture," IEEE Trans. Circuits & Syst. II, vol.40, no.11, pp.727–729, Nov. 1993.
[9] K. Matsubara, K. Nishikawa, and H. Kiya, "Pipelined adaptive filters based on delayed LMS algorithm," Trans. of IEICE, vol.J79-A, no.5, pp.1050–1057, May 1996.
[10] M. Ohki and S. Hashiguchi, "Two-dimensional LMS adaptive filters," IEEE Trans. CE, vol.37, no.1, pp.66–73, Feb. 1991.
[11] Y. Nakamura, K. Oguri, A. Nagoya, M. Yukishita, and R. Nomura, "High-level synthesis design at NTT systems labs," IEICE Trans. Inf. & Syst., vol.E76-D, no.9, pp.1047–1054, Sept. 1993.

Katsushige Matsubara was born in Kanagawa, Japan, on October 16, 1971. He received the B.E. degrees in electronics and information engineering from Tokyo Metropolitan University in 1995. He is currently a candidate for the M.E. degree at Tokyo Metropolitan University. His research interests are in adaptive signal processing.



Kiyoshi Nishikawa was born in Tokyo, Japan, on December 12, 1966. He received the B.E., the M.E., and the D.E. degrees in electrical engineering from Tokyo Metropolitan University in 1990, 1992 and 1996, respectively. From 1992 to 1993, he was at the Computer Systems Laboratory, Nippon Steel Corp. as a researcher. Since 1993, he has been with Tokyo Metropolitan University as an Assistant Professor. His research interest includes the adaptive signal processing. He is a member of IEEE SP, CAS, IT, and Computer Society.



Hitoshi Kiya was born in Yamagata, Japan, on November 16, 1957. He received the B.E. and M.E. degrees in electrical engineering from Nagaoka University of Technology, Niigata, Japan, and the D.E. degree in electrical engineering from Tokyo Metropolitan University, Tokyo, Japan, in 1980, 1982, and 1987, respectively. In 1982, he joined Tokyo Metropolitan University, where he is currently an Associate Professor of Electronics and Information Engineering. His research interests are in digital signal processing, multirate systems, adaptive filtering, image processing, and efficient algorithms for VLSI implementation. Dr. Kiya is a Member of the Institute of Electrical and Electronics Engineers, Inc. (IEEE) of USA, the Image Electronics Engineers of Japan and the Institute of Television Engineers of Japan.