PAPER *Special Section on VLSI for Digital Signal Processing*

A Pipelined Architecture for Normalized LMS Adaptive Digital Filters

Akio HARADA[†], ***Student Member***, **Kiyoshi NISHIKAWA**[†], ***and*** **Hitoshi KIYA**[†], ***Members***

SUMMARY A pipelined architecture is proposed for the normalized least mean square (NLMS) adaptive digital filter (ADF). Pipelined implementation of the NLMS has not yet been proposed. The proposed architecture is the first attempt to implement the NLMS ADF in the pipelined fashion. The architecture is based on an equivalent expression of the NLMS derived in this study. It is shown that the proposed architecture achieves a constant and a short critical path without producing output latency. In addition, it retains the advantage of the NLMS, i.e., that the step size that assures the convergence is determined automatically. Computer simulation results that confirm that the proposed architecture achieves convergence characteristics identical to those of the NLMS.

key words: *adaptive digital filter, pipeline processing, normalized LMS algorithm*

1. Introduction

In this paper, we propose a pipelined architecture for the adaptive digital filters (ADFs) based on the normalized least mean square (normalized LMS: NLMS) algorithm. The proposed architecture is the first attempt to implement the NLMS ADF in the pipelined fashion. It retains the advantage of the NLMS, which is that the value of the step-size parameter is determined automatically, thereby assuring convergence.

Pipelined processing of gradient-type ADFs has been considered [1]–[4] as a possible way of meeting requirements for processing signals at very high speed or at low power consumption. Several architectures [2]–[4] have been proposed so far that are based on the delayed LMS (DLMS) [5], [6] algorithm. The DLMS is obtained by inserting the delays into the error feedback loop of the LMS, and its hardware implementation is achieved by retiming these delays [7].

In implementing these conventional architectures, a difficulty arises in determining the value of the step size. The convergence characteristic of the original LMS is determined by the step size parameter μ [8]. The value of μ must be chosen beforehand, based on the statistical characteristics of the input signals. But this is a difficult task in some applications where the characteristics of the input are unknown or vary with time. Additionally, the delay in the DLMS makes the task even more difficult, because the inserted delays affect the input-output relation of the LMS algorithm [5], [6]. This may create a problem when the algorithm is implemented in VLSI based on conventional architectures.

The NLMS [9] is known as an improved version of the LMS. In the NLMS, the value for μ, which assures convergence, is determined automatically based on the Euclidean norm of the input vector. The hardware implementation of the pipelined NLMS, however, has not yet been considered. Because the delays in the DLMS affect the input-output relation of the LMS, and therefore, the condition for convergence is also affected. Therefore, even if the step size is normalized, the convergence of the algorithm cannot be ensured.

In this paper, we propose a pipelined architecture for NLMS ADFs. The proposed architecture is constructed by using an equivalent expression of the NLMS which is derived in this paper. We show that the proposed architecture has a constant and a short critical path. Moreover, it does not produce output latency.

This paper is organized as follows. In Sect. 2, a review of the conventional methods for pipelining gradient-type ADFs is given. In Sect. 3, the equivalent expression of the NLMS is proposed. Then, in Sect. 4, we propose an architecture based on the derived expression, and compare it with the conventional one, and present the simulation results. Conclusions are given in Sect. 5.

2. Conventional Pipelined Architecture for Gradient-Type ADFs

In this section, we describe the pipelined architectures based on the DLMS as conventional pipelined architecture for gradient-type ADFs and their problems.

2.1 Pipelined Architectures Based on the DLMS

The filter update-formula of the DLMS algorithm is given as [5], [6]

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \mu e(n-D_d)\mathbf{U}(n-D_d) \quad (1)$$

$$e(n-D_d) = d(n-D_d) - \mathbf{W}^T(n-D_d)\mathbf{U}(n-D_d), \quad (2)$$

where $\mathbf{W}(n)$ and $\mathbf{U}(n)$ are the N adaptive filter coefficients and the N input samples at time n, and they

Manuscript received June 22, 1998.
Manuscript revised September 9, 1998.

[†]The authors are with the Department of Electrical Engineering, Graduate School of Tokyo Metropolitan University, Hachioji-shi, 192–0397 Japan.

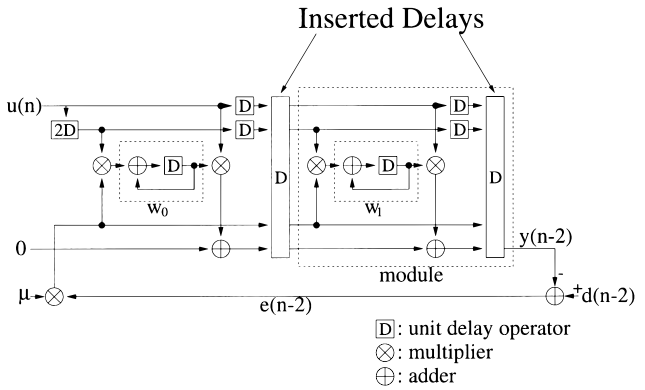

Fig. 1 Signal flow graph for the pipelined DLMS ADF ($N = 2$).

are given as

$$\mathbf{W}(n) = [w_0(n), w_1(n), \cdots, w_{N-1}(n)]^T \quad (3)$$

$$\mathbf{U}(n) = [u(n), u(n-1), \cdots, u(n-N+1)]^T \quad (4)$$

respectively; $d(n)$ and $e(n)$ are the desired and the error signals at time n; and μ is the step size. In (1), D_d is the number of delays inserted into the error feedback path.

The pipelined implementations [2]–[4] of the DLMS ADF are achieved by replacing the delays D_d using the retiming technique [7], so that a shorter critical path can be achieved as D_d increases.

2.2 Problems of the Pipelined DLMS ADF

The pipelined architecture based on the DLMS has two serious problems. One is concerning the value of the step-size parameter: Although a shorter critical path can be achieved as D_d increases, at the same time, increasing of D_d narrows the selectable range for μ, namely, the convergence characteristic becomes poor as D_d increases [5], [6].

There is another problem that the output latency proportional to the number of inserted delays will be produced. Let us show this fact using an example under the condition $N = D_d = 2$. The output is delayed by one clock cycle per delay as shown in Fig. 1. In this case, the output is delayed for two clock cycles.

As will be described in the next section, the NLMS is obtained by normalizing the step size of the LMS by the squared Euclidean norm of the input vector. However, in the DLMS, the inserted delays affect the input-output relation of the LMS, and, therefore, the condition for convergence. This means that even if the step size is normalized, the DLMS cannot be assured to converge.

3. Equivalent Expression of the NLMS

Here, we derive an equivalent expression of the NLMS algorithm on which the proposed architecture will be constructed.

3.1 NLMS Algorithm

First, the filter update-formula of the NLMS algorithm is given as [8]

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \frac{\alpha e(n)}{\| \mathbf{U}(n) \|^2}\mathbf{U}(n) \quad (5)$$

$$e(n) = d(n) - \mathbf{W}^T(n)\mathbf{U}(n), \quad (6)$$

where $\| \mathbf{U}(n) \|$ shows the Euclidean norm of the input vector $\mathbf{U}(n)$. A parameter α is for adjusting both the convergence and misadjustment, and its value should be in the range $0 < \alpha < 2$ to ensure the convergence [8]. Note that α does not depend on $\mathbf{U}(n)$ theoretically.

The pipelined implementation cannot be achieved by using the filter updete-formula (5), (6). We should consider a transformation of (5), (6) to implement pipelined ADFs.

3.2 Look-Ahead Transformation of the NLMS Algorithm

By applying the *n-step look-ahead transformation* of (6) [1], we obtain

$$e(n) = d(n) - y(n), \quad (7)$$

where $y(n)$ is given as

$$y(n) = \left\{ \sum_{k=1}^{n} \frac{\alpha e(n-k)}{\| \mathbf{U}(n-k) \|^2}\mathbf{U}^T(n-k) \right\} \mathbf{U}(n). \quad (8)$$

Note that we assumed the initial condition $\mathbf{W}(0) = \mathbf{0}$.

The update-formula itself is not affected by the *look-ahead transformation*. Therefore, the selectable range for α is identical to that of the NLMS.

3.3 Pipelining of Calculation of $y(n)$

Here we show that the pipelined implementation of the NLMS can be achieved by using the above expressions. The NLMS ADF would be pipelined if the calculation of $y(n)$ can be pipelined.

We expand (8) in a scalar form as

$$y(n) = \sum_{k=1}^{n} \sum_{m=0}^{N-1} \frac{\alpha e(n-k)u(n-k-m)u(n-m)}{\| \mathbf{U}(n-k) \|^2}. \quad (9)$$

By defining new variables $h_i(n)$ and $r_i(n)$ as

$$h_i(n) = \sum_{j=2}^{n-i} \frac{\alpha e(n-j)u(n-j-i)}{\| \mathbf{U}(n-j) \|^2} \quad (10a)$$

$$r_i(n) = \begin{cases} \sum_{j=1}^{N-i-1} u(n-j-i-1)u(n-j) & (0 \le i < N-1) \\ 0 & (i = N-1) \end{cases} \quad (10b)$$

(9) is expressed as

$$y(n) = \sum_{i=0}^{N-1} \left[\{h_i(n-i)u(n-i) + \{r_i(n-i) + u(n-2i-1)u(n-i)\} \frac{\alpha e(n-i-1)}{\| \mathbf{U}(n-i-1) \|^2} \right]. \quad (11)$$

A brief derivation of (11) is given in Appendix A. Using this form, we show that the calculation of $y(n)$ and hence, the NLMS ADF can be pipelined.

We define $c_i(n)$ as

$$c_i(n) = h_i(n)u(n) + \{r_i(n) + u(n-i-1)u(n)\} \cdot \frac{\alpha e(n-1)}{\| \mathbf{U}(n-1) \|^2}. \quad (12)$$

Then, (11) is rewritten in a single summation form as

$$y(n) = \sum_{i=0}^{N-1} c_i(n-i). \quad (13)$$

By defining $y_i(n)$ as

$$y_i(n) = \begin{cases} y_{i+1}(n-1) + c_i(n) & (0 \le i < N) \\ 0 & (i = N) \end{cases} \quad (14)$$

we have the following relation:

$$y(n) = y_0(n). \quad (15)$$

Known from the form of Eq. (14), the calculation of $y_i(n)$ can be regarded as a processing-module. By connecting N modules in series, we can implement the NLMS ADF in a pipelined fashion.

At this point, we must notice that each processing module $y_i(n)$ is composed of $h_i(n)$ and $r_i(n)$ instead of the filter coefficient $w_i(n)$. This is the major difference from the conventional architectures based on the DLMS ADF [2]–[4] and this difference enables us to implement the pipelined-processing of the NLMS.

Note that we can calculate $h_i(n)$, $r_i(n)$ and $\| \mathbf{U}(n) \|^2$ recursively as

$$h_i(n) = h_i(n-1) + \frac{\alpha e(n-2)u(n-i-2)}{\| \mathbf{U}(n-2) \|^2} \quad (16a)$$

$$r_i(n) = r_i(n-1) + u(n-i-2)u(n-1) - u(n-N-1)u(n-N+i) \quad (16b)$$

$$\| \mathbf{U}(n) \|^2 = \| \mathbf{U}(n-1) \|^2 + u(n)^2 - u(n-N)^2 \quad (16c)$$

and the use of these formulas greatly reduces the number of calculators. Thus, we have seen that we can achieve the pipelined implementation of the NLMS ADF by using $h_i(n)$ and $r_i(n)$ instead of using $w_i(n)$.

4. Proposed Architecture

Here, we propose a pipelined architecture based on the expression described in Sect. 3.

4.1 Proposed Architecture

The proposed architecture is depicted in Fig. 2. As is shown in the figure, the proposed architecture is composed of a series of N modules; each module calculates $c_i(n)$. The structures of the i-th processing module is shown in Fig. 3. Besides, the module for calculating $\alpha / \| \mathbf{U}(n) \|^2$ is shown in Fig. 4 which implements (16c) and calculation of dividing α by $\| \mathbf{U}(n) \|^2$.

4.2 Output Latency and Critical Path

Here, let us consider the output latency and the critical path of the proposed architecture. In the proposed architecture, there exists no delay between the input $u(n)$ and the output $y(n) = y_0(n)$ as known from Figs. 2 and 3. Therefore, the output latency is guaranteed to be always 0.

Then, the critical path t_{cp} is given as

$$t_{cp} = \begin{cases} 2t_{mult} + 2t_{add} & (t_{div} \le t_{mult} + t_{add}) \\ t_{div} + t_{mult} + t_{add} & (t_{mult} + t_{add} < t_{div}) \end{cases} \quad (17)$$

where t_{div}, t_{mult} and t_{add} are the times required for one division, one multiplication and one addition respectively. Note that in the 0-th processing module, we use the equation

$$\begin{aligned} y_0'(n) &= d(n) - y_0(n) \\ &= d(n) - (y_1(n-1) + c_0(n)) \\ &= e(n), \end{aligned} \quad (18)$$

instead of (14).

5. Comparison and Simulation

In this section, we compare the proposed architecture with the non-pipelined NLMS and the conventional architecture. We show, through simulation results, the effectiveness of the proposed architecture.

Table 1 Comparison of each architecture under the condition $t_{div} \leq t_{mult} + t_{add}$ where N is filter length, t_{div}, t_{mult} and t_{add} are the times required for one division, one multiplication and one addition.

Architecture	Automatic determination	Critical path	Output latency	Number of operators			
				Multiplier	Divider	Adder	Delay
non-pipelined NLMS	◯	$3t_{mult} + (N+2)t_{add}$	0	$2N+3$	1	$2N+3$	$2N$
conventional (LMS) [10]	×	$t_{mult} + 2t_{add}$	N	$5N+1$	0	$5N+1$	$6N$
proposed	◯	$2t_{mult} + 2t_{add}$	0	$5N+2$	1	$5N+3$	$5N$

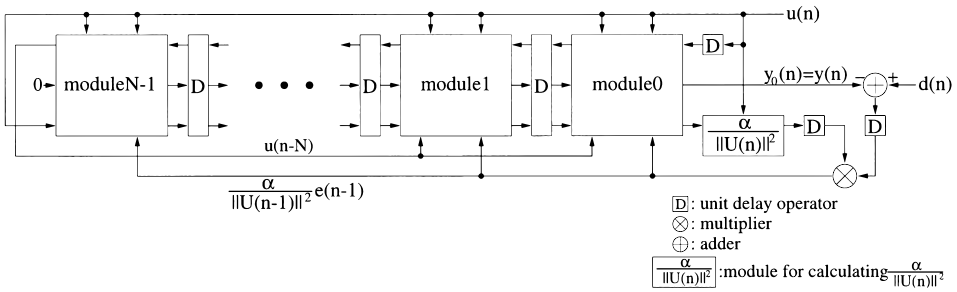

Fig. 2 Pipelined architecture for the NLMS ADF.

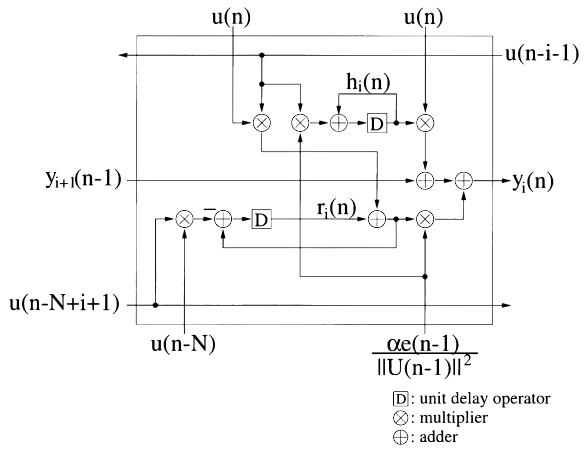

Fig. 3 Structure of the i-th processing module.

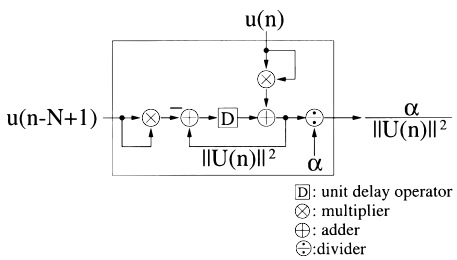

Fig. 4 Structure of the module for calculating $\alpha/\| \mathbf{U}(n) \|^2$.

5.1 Comparison

The pipelined architecture of the NLMS has not been proposed yet as mentioned in introduction. Therefore, the proposed architecture is compared with the pipelined architecture [10] of the LMS as the conventional architecture. Although several pipelined architecture of the LMS had been proposed [10], [11], we only use the architecture of [10] in the following comparison. This selection was done in order to make the comparison simple, because these architectures have similar property.

They are based on the DLMS, and by adding the difference between the LMS and the DLMS as a correction term to the error signal, they achieve the equivalent convergence characteristics with that of the LMS. However, the problem of producing the output latency cannot be solved. Therefore, even if $\| \mathbf{U}(n) \|^2$ is set as $\| \mathbf{U}(n) \|^2 = 1$ in the proposed architecture, the proposed architecture is different from the conventional architectures [10], [11].

We compare the proposed architecture with the non-pipelined NLMS ADF and the conventional one in terms of the automatic determination of the value of the step-size parameter, the critical path, the output latency, number of broadcast lines and the number of operators. We show the comparison in Table 1.

We can see from the table, the proposed architecture achieves a shorter critical path, which is independent of the filter length N, than the non-pipelined NLMS. However the critical path of the proposed architecture is slightly longer than that of [10]. On the other hand, the proposed architecture is better than [10] in that it does not produce output latency. Moreover the proposed architecture has the advantage that it can automatically determine the value of the step-size parameter: this feather can improve the stability of the algorithm in the practical applications.

Note that the proposed architecture has 3 broadcast lines, and these lines may cause a problem when the architecture would be implemented. Besides we notice that the pipelined architecture for the DLMS depicted in Fig. 1 has no broadcast line, but the con-

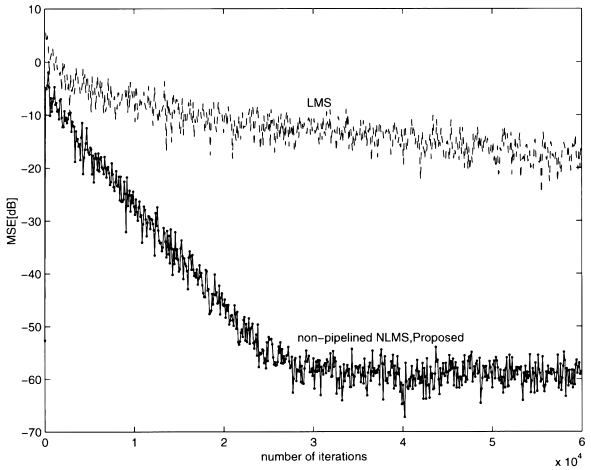

Fig. 5 Convergence characteristics of the proposed architecture, the NLMS ADF and the LMS ADF when the input signal was the colored process.

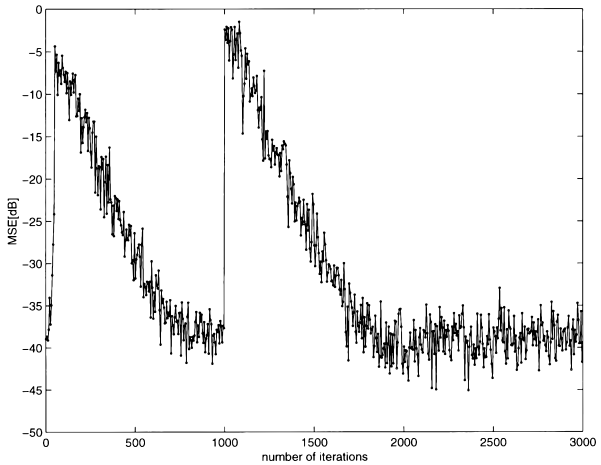

Fig. 6 Convergence characteristics of the proposed architecture and the NLMS ADF when the input signal was the white process and the unknown system was non stationary.

ventional architecture [10] has 3 broadcast lines.

Although an extraction of the filter coefficients $\mathbf{W}(n)$ from the DLMS ADF on learning process is easy, in the case of the proposed architecture, it may be difficult. However, the filter coefficients can be reconstructed after learning process is finished, because the proposed architecture is derived from the equivalent expression of the NLMS.

5.2 Simulation

Finally we show simulation results to demonstrate the effectiveness of the proposed architecture. We simulated a system identification problem with an unknown system of the filter length 100 with 2 different conditions.

(1) The input signal was an AR(1) process with the AR parameter 0.99 as a colored signal, and the unknown system was stationary.
(2) The input signal was a zero-mean Gaussian random process with a variance of 1 as a white process, the unknown system was non stationary, namely, when the number of iterations becomes 1000, the unknown system was changed.

The observation noise signal was a zero-mean Gaussian random process with a variance of 0.0001 which is independent of the input. The results are ensemble averages of ten independent processes.

First, we show the results under the condition (1). In this case, we compared the proposed architecture, the NLMS ADF and the LMS ADF. We set the filter length N of ADFs as $N = 100$. The step size μ and the parameter α were set as $\mu = 0.00002$ for the LMS ADF, and $\alpha = 1.0$ for the others. The value of μ was selected as the value which achieves the fastest rate of convergence after some trials with different value of μ.

The simulation results are shown in Fig. 5. From the figure, we can see that the proposed architecture realizes the identical convergence characteristic with that of the NLMS ADF.

Then, we show the results under the condition (2). In the case, we compared the proposed architecture and the NLMS ADF. We set the filter length N of ADFs as $N = 200$. the parameter α was set as $\alpha = 1.0$ for both. The simulation results are shown in Fig. 6. From the figure, we can see that the proposed architecture can realize the identical convergence characteristic with that of the NLMS ADF as in the stationary unknown system case.

6. Conclusion

In this paper, we proposed a pipelined architecture of the NLMS ADF which was first proposal so far. The proposed architecture is based on an equivalent expression of the NLMS algorithm which was derived in this paper. We also showed that the proposed architecture can achieve pipelined processing with the identical convergence characteristic as that of the NLMS ADF, without producing output latency.

References

[1] N.R. Shanbhag and K.K. Parhi, "Pipelined Adaptive Digital Filters," Kluwer Academic Publishers, 1994.
[2] K. Matsubara, K. Nishikawa, and H. Kiya, "Pipelined adaptive filters based on delayed LMS algorithm," IEICE Trans., vol.J79-A, pp.1050–1057, May 1996.
[3] 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 Processing, vol.40, no.11, pp.2799–2803, Nov. 1992.
[4] 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.
[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 Processing, vol.40, no.1, pp.230–232, Jan. 1992.
[7] C.E. Leiserson, F. Rose, and J. Saxe, "Optimizing synchronous circuitry by retiming," Proc. Third Caltech Conference on VLSI, pp.87–116, March 1983.
[8] S. Haykin, "Adaptive Filter Theory," Prentice-Hall, Inc., 1991.
[9] J. Nagumo and A. Noda, "A learning method for system identification," IEEE Trans. Autom. Control, pp.282–287, June 1967.
[10] Q. Zhu, S.C. Douglas, and K.F. Smith, "A pipelined architecture for LMS adaptive FIR filters without adaptation delay," Proc. IEEE ICASSP '97, vol.III, pp.1933–1936, April 1997.
[11] K. Matsubara, K. Nishikawa, and H. Kiya, "A new pipelined architecture of the LMS algorithm without degradation of convergence characteristics," Proc. IEEE ICASSP '97, vol.III, pp.4125–4128, April 1997.

Appendix A: Derivation of (11)

This appendix provides a brief derivation of (11).

Let us consider the NLMS ADF with 3 coefficients. The output $y(n)$ is given as

$$\begin{aligned} y(n) &= \mathbf{W}^T(n)\mathbf{U}(n) \\ &= w_0(n)u(n) + w_1(n)u(n-1) \\ &\quad + w_2(n)u(n-2). \end{aligned} \tag{A·1}$$

By applying *n-step look-ahead transformation* to (A·1), we obtain

$$\begin{aligned} y(n) = &\sum_{k=1}^{n} \frac{\alpha e(n-k)u(n-k)u(n)}{\| \mathbf{U}(n-k) \|^2} \\ &+ \sum_{k=1}^{n} \frac{\alpha e(n-k)u(n-k-1)u(n-1)}{\| \mathbf{U}(n-k) \|^2} \\ &+ \sum_{k=1}^{n} \frac{\alpha e(n-k)u(n-k-2)u(n-2)}{\| \mathbf{U}(n-k) \|^2}. \end{aligned} \tag{A·2}$$

Note that the initial condition $\mathbf{W}(0) = \mathbf{0}$.

From (A·2), we find that the output $y(n)$ is given as a sum of the products of $\{u(n-j) : 0 \le j\}$ and $\{e(n-k)/\| \mathbf{U}(n-k) \|^2 : 1 \le k\}$. Equation (A·2) can be divided into two parts: the first part consists of the terms which cannot be calculated before the arrival of $u(n)$ or $e(n-1)/\| \mathbf{U}(n-1) \|^2$ at time n, and the second one consists of those which can be calculated without $u(n)$ and $e(n-1)/\| \mathbf{U}(n-1) \|^2$. Namely, the first part is given as

$$\begin{aligned} c_0(n) = &\sum_{j=2}^{n} \frac{\alpha e(n-j)u(n-j)u(n)}{\| \mathbf{U}(n-j) \|^2} \\ &+ \sum_{j=0}^{2} \frac{\alpha e(n-1)u(n-j-1)u(n-j)}{\| \mathbf{U}(n-1) \|^2}, \end{aligned} \tag{A·3}$$

and the second part is given as

$$\begin{aligned} c'_0(n) &= y(n) - c_0(n) \\ &= \sum_{k=2}^{n} \frac{\alpha e(n-k)u(n-k-1)u(n-1)}{\| \mathbf{U}(n-k) \|^2} \\ &\quad + \sum_{k=2}^{n} \frac{\alpha e(n-k)u(n-k-2)u(n-2)}{\| \mathbf{U}(n-k) \|^2}. \end{aligned} \tag{A·4}$$

The second part $c'_0(n)$ can be calculated at time $n-1$, because $c'_0(n)$ does not require the knowledge of $u(n)$ and $e(n-1)/\| \mathbf{U}(n-1) \|^2$.

In the similar way, $c'_0(n)$ can be divided into two parts $c_1(n-1)$ and $c_2(n-2)$ defined as

$$\begin{aligned} &c_1(n-1) \\ &= \sum_{j=2}^{n-1} \frac{\alpha e(n-1-j)u(n-2-j)u(n-1)}{\| \mathbf{U}(n-1-j) \|^2} \\ &\quad + \sum_{j=0}^{1} \frac{\alpha e(n-2)u(n-3-j)u(n-1-j)}{\| \mathbf{U}(n-2) \|^2}, \end{aligned} \tag{A·5}$$

$$\begin{aligned} &c_2(n-2) \\ &= \sum_{j=2}^{n-2} \frac{\alpha e(n-2-j)u(n-4-j)u(n-2)}{\| \mathbf{U}(n-2-j) \|^2} \\ &\quad + \frac{\alpha e(n-3)u(n-5)u(n-2)}{\| \mathbf{U}(n-3) \|^2}, \end{aligned} \tag{A·6}$$

$c_1(n-1)$ can be calculated using the information available at time $n-1$. Using (A·3), (A·5) and (A·6), $y(n)$ can be expressed as

$$\begin{aligned} &y(n) \\ &= \sum_{i=0}^{2} c_i(n-i) \\ &= \sum_{i=0}^{2} \left[\sum_{j=2}^{n-i} \frac{\alpha e(n-i-j)u(n-2i-j)u(n-i)}{\| \mathbf{U}(n-i-j) \|^2} \right. \\ &\quad \left. + \sum_{j=0}^{2-i} \frac{\alpha e(n-i-1)u(n-2i-j-1)u(n-i-j)}{\| \mathbf{U}(n-i-1) \|^2} \right]. \end{aligned} \tag{A·7}$$

By using $h_i(n)$ and $r_i(n)$ shown in (10a,b), we obtain (11) under the condition $N = 3$ as

$$y(n) = \sum_{i=0}^{2} \left[\{h_i(n-i)u(n-i) + \{r_i(n-i) + u(n-2i-1)u(n-i)\} \frac{\alpha e(n-i-1)}{\| \mathbf{U}(n-i-1) \|^2} \right]. \quad (\mathrm{A}\cdot 8)$$

Akio Harada was born in Tokyo, Japan, on October 31, 1974. He received the B.E. degree in electronics and information engineering from Tokyo Metropolitan University in 1997. He is currently a candidate for the M.E. degree at Tokyo Metropolitan University. His research interest is 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 a research associate. His research interest includes the adaptive signal processing. He is a member of IEEE SP, CAS, COM, and Computer Societies.

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 Electrical Engineering, Graduate School of Engineering. He was a visiting researcher of the University of Sydney in Australia from Oct. 1995 to Mar. 1996. His research interests are in digital signal processing, multirate systems, adaptive filtering, image processing, and efficient algorithms for VLSI implementation. He is an Associate Editor of Trans. on Signal Processing of IEEE and Trans. of IEICE. He is a Member of the IEEE, the Image Electronics Engineers of Japan and the Institute of Television Engineers of Japan.