# Design and Implementation of Multiplierless Adjustable Fractional-Delay All-Pass Filters

Juha Yli-Kaakinen and Tapio Saramäki
Institute of Signal Processing
Tampere University of Technology
P. O. Box 553, FI-33101 Tampere, Finland
email: {ylikaaki, ts}@cs.tut.fi

***Abstract*— This paper describes an algorithm for finding the multiplierless coefficient representations for adjustable fractional-delay (AFD) all-pass filters. The optimization is performed in three basic steps. First, an initial filter is generated using a simple design scheme. Second, this filter is used as a start-up solution for the nonlinear optimization algorithm which is employed for determining a parameter space of the infinite-precision coefficients. This space includes the feasible space where the filter meets the given criteria. The third step involves finding the discrete coefficient values in this space so that the resulting filter meets the criteria with the simplest coefficient representation forms. Examples are included for illustrating the efficiency of the proposed synthesis scheme. In addition, the performance and the complexity of the multiplierless AFD all-pass filters are compared with those of the multiplierless AFD finite-impulse response filters implemented using the modified Farrow structure proposed by Vesma and Saramäki. This comparison shows that the number of adders for the resulting filters are in the best case less than 50 percent compared with those implemented using the modified Farrow structure.**

## I. Introduction

In various digital signal processing applications, there is a need for a delay that is a fraction of the sampling interval. Furthermore, it is often desired that the delay value is adjustable or variable during the computation. These applications include, e.g., sampling rate conversion, echo cancellation, phased array antenna systems, time delay estimation, timing adjustment in all-digital receivers, modeling of musical instruments, and speech coding and synthesis [1], [2].

Adjustable fractional-delay (AFD) filters can be designed either using finite-impulse response (FIR) or infinite-impulse response (IIR) filters. One computationally efficient technique, belonging to the former filter class, is to use the Farrow structure [1] consisting of several parallel fixed FIR filters. The desired fractional delay is achieved by properly multiplying the outputs of these filters with quantities depending directly on the value of the fractional delay [1]–[5]. Another attractive class of AFD filters, belonging to the second class, are AFD all-pass filters based on so-called gathering structure [2], [6]–[9] proposed by Makundi, Laakso, and Välimäki. In this structure, the filter coefficients are the polynomials of the fractional delay parameter.

The main advantage of the AFD filters based on all-pass structures is that the magnitude response of all-pass filters is equal to unity at all the frequencies [6]–[8], whereas for the AFD FIR filters the structure does not enable one to approximate unity at all the frequencies. Further, the overall delay of an IIR structure satisfying the same phase delay criteria is, especially in cases requiring stringent magnitude specifications, considerably smaller than its FIR filter counterpart. Moreover, the complexity of the AFD filters, that is, the number of multipliers, adders, and delays, required to implement the overall filter, is significantly smaller for the AFD all-pass filters compared with their FIR counterparts [8].

The main disadvantage of the AFD all-pass filters compared to the AFD FIR filters is their higher roundoff noise level at the filter output. In addition, because of the recursive nature of IIR filters, the abrupt changes in their parameters may cause transients at the filter output. Further, the design of the AFD all-pass filters is more complicated due to stability issues being inherent when synthesizing IIR filters.

The purpose of this contribution is to propose an algorithm for finding the multiplierless coefficient representations for the AFD all-pass filters. This algorithm is based on the following observation: Finding the smallest and largest values of the filter coefficients in such a way that the given criteria are still met enables one to find a parameter space including the feasible space where the filter specifications are met. After finding this larger space, all what is needed is to check whether in this space there exist the desired discrete values for the coefficient representations. In addition, the complexity of the multiplierless AFD filters based on the use of all-pass structure is compared, by means of examples taken from the literature, with their FIR counterparts implemented using the modified Farrow structure proposed by Vesma and Saramäki. This comparison shows that the number of adders and delays for the resulting filters are in the best case 45 and 25 percent, respectively, compared with those implemented using the modified Farrow structure.

## II. AFD All-Pass Filters

Let the desired frequency response of an AFD filter be

$$H_{\rm des}(e^{j\omega},\,\mu) = e^{-j\omega(D+\mu)}, \tag{1}$$

where $D$ is a fixed integer delay and $\mu$ is an adjustable fractional delay in the range $[-1,\,0]$.[1] The corresponding ideal phase response is given by

$$\arg H_{\rm des}(e^{j\omega},\,\mu) = -\omega(D+\mu). \tag{2}$$

The transfer function of the AFD all-pass filter [2], [7]–[9] chosen to approximate $H_{\rm des}(e^{j\omega},\,\mu)$ is expressible as

$$H_A(z,\,\mu) = \frac{z^{-N}A(z^{-1},\,\mu)}{A(z,\,\mu)}, \tag{3a}$$

where the denominator polynomial is given by

$$A(z,\,\mu) = 1 + \sum_{n=1}^{N} b_n(\mu) z^{-n} = 1 + \sum_{n=1}^{N}\left(\sum_{p=1}^{P} c_{pn}\mu^p\right) z^{-n}, \tag{3b}$$

that is, each coefficient in the overall filter is given as a polynomial functions of degree $P$ in $\mu$. The phase delay response of this all-pass filter is expressible as

$$\tau_A(\omega,\,\mu) = N - \frac{2}{\omega}\arctan\left(\frac{\sum_{n=1}^{N} b_n(\mu)\sin n\omega}{1+\sum_{n=1}^{N} b_n(\mu)\cos n\omega}\right). \tag{4}$$

The above structure does not enable one to keep the phase delay response, as given by Eq. (4), within the limits $N+\mu\pm\delta_p$ in the overall baseband $[0,\,\pi]$. This is because the phase response of a stable all-pass

[1] Traditionally, when designing filters with an adjustable fractional delay, the delay is expressed as a sum of an integer and a positive fractional delay. However, for filters under consideration in this contribution it has turned out to be more beneficial, from the implementation point of view, to express the overall delay to be an integer and a negative of the fractional delay. If it is desired to use the common approach, then $D$ in Eq. (1) should be replaced by $D-1$ and $\mu$ by $1+\mu$. It is worth emphasizing that if the traditional approach is used, then the resulting overall structure becomes more complicated.

0-7803-8834-8/05/$20.00 ©2005 IEEE.

filter decreases monotonously from 0 to $-N\pi$ as $\omega$ increases from 0 to $\pi$ [10]. Therefore, this contribution concentrates on approximating the desired phase response on the frequency band given by $\Omega_p = [0, \omega_p]$, where $\omega_p < \pi$. The goal is to determine the discrete-valued representation forms for the adjustable parameters such that for each value of $\mu$ within $-1 \leq \mu \leq 0$, the phase delay closely approximates $N + \mu$.

## III. Problem Statement

Before stating the optimization problem, the phase delay response of the filter is denoted by $\tau_A(\Phi, \omega, \mu)$, where $\Phi$ is the following adjustable parameter vector:

$$\Phi = [c_{11}, \ldots, c_{1N}, c_{21}, \ldots, c_{2N}, \ldots, c_{P1}, \ldots, c_{PN}]. \quad (5)$$

Given the passband edge $\omega_p$ and the passband ripple $\delta_p$ the phase delay specifications for the filter are stated as follows:

$$\max_{-1\leq\mu\leq 0}[\max_{\omega\in\Omega_p}|\tau_A(\Phi, \omega, \mu) - (N + \mu)|] \leq \delta_p. \quad (6)$$

When the above criteria are met, then for each value of $\mu$, the phase delay response stays within the limits $N + \mu \pm \delta_p$ on $\Omega_p$.

In order to guarantee that the resulting filter is stable, it is required that the poles of the transfer function are inside the unit circle for each value of $\mu$ within $-1 \leq \mu \leq 0$. The stability can be easily checked by using Schür-Cohn stability test [10]. This test gives for an $N$th-order denominator polynomial the stability test parameters $k_i$ for $i = 1, 2, \ldots, N-1$. For the stability, it is required that the maximum absolute value of the $k_i$'s is smaller than unity. It has turned out that for the AFD all-pass filters, the largest pole radius is achieved when $\mu = -1$. Therefore, for the stability it is required that for the corresponding denominator polynomial $A(z, -1)$, the stability test parameters satisfy the condition

$$|k_i| < 1 \quad \text{for } i = 1, 2, \ldots, N-1. \quad (7)$$

In many hardware or very-large-scale-integration (VLSI) implementations, it is attractive to carry out the multiplication of a data sample by a filter coefficient value using a sequence of shifts and adds or subtracts. For such a purpose, it is desired to express the coefficient values in the form

$$\sum_{r=1}^{R} a_r 2^{-S_r}, \quad (8)$$

where each $a_r$ is either 1 or $-1$ and the $S_r$'s are nonnegative integers in the increasing order. In this case, the goal is to find all the coefficient values so that: 1) $R$, the number of powers-of-two terms, is made as small as possible, 2) $S_R$, the maximum number of shifts, is made as small as possible. For this purpose, it is attractive to use the canonic-signed-digit (CSD) representation. This representation is characterized by the fact that no two consecutive digits $a_r$ are both nonzero, that is, for the minimal $R$, $a_r a_{r+1} = 0$ for $r = 1, 2, \ldots, R-1$. In the sequel, $\mathbb{CSD}_{(R,\, S_R)}$ denotes the space of the CSD numbers with the maximum number of power-of-two terms and the maximum number of fractional bits being $R$ and $S_R$, respectively.

The implementation cost is the number of adders and subtracters required to implement all the filter coefficients as well as the structural adders required to implement the filter structure, that is, the implementation cost is given by

$$N(P + 1) + \sum_{n=1}^{N}\sum_{p=1}^{P} \sigma_{pn}, \quad (9)$$

where $\sigma_{pn}$ for $p = 1, 2, \ldots, P$ and $n = 1, 2, \ldots, N$ is the number of adders and subtracters required to implement $c_{pn}$. The optimization problem under consideration is the following:

*Optimization Problem:* Given $\omega_p$ and $\delta_p$, find $N$, $P$, and the parameter vector $\Phi$ in such a manner that, first, the criteria of Eqs. (6) and (7) are met after quantizing the values of the filter coefficients corresponding to the parameters included in $\Phi$ to achieve the above-mentioned form for their representations and, then, the implementation cost, as given by Eq. (9), is minimized.

## IV. Proposed Three-Step Design Scheme

The solutions to the stated optimization problem can be found in the following three steps. In the first step, an initial filter is designed using a simple design scheme. In the second step, the smallest and largest values are determined for each adjustable parameter by reoptimizing the remaining unknowns in a parameter vector in such a manner that the given specifications are met. This enables one to find the parameter space of the infinite-precision coefficients including the feasible space where the filter meets the specifications. The third step involves finding the filter parameters in this space such that the resulting filter meets the given criteria with the simplest coefficient representation forms.

### A. Algorithm for Finding an Initial Infinite-Precision Filter

The convergence of the infinite-precision optimization algorithm to be described in the following subsection to the optimal solutions implies a rather good initial solution for the adjustable parameters. An initial solution for further optimization can be derived by slightly modifying the optimization algorithm proposed by Laakso *et al.* in [2]. In this algorithm, the weighted least-squares phase delay error is minimized for a set of $\mu$'s and based on the resulting coefficient sets a polynomial approximation is derived for the coefficients. For more details, see [9].

### B. Optimization of Infinite-Precision Filter

It has turned out that a very straightforward optimization scheme for finding the parameter space is obtained as follows: For each filter coefficient, the smallest and largest values are determined so that by reoptimizing the values of the remaining coefficients the given overall criteria, as given by Eqs. (6) and (7), can still be met. This goal is achieved by solving $2NP$ problems of the following form. Find the adjustable parameter vector $\Phi$ to minimize $\psi$ subject to the conditions of Eqs. (6) and (7). Here, $\psi$ is $-c_{pn}$ or $c_{pn}$ where $c_{pn}$ is one among the filter coefficients $c_{pn}$ for $p = 1, 2, \ldots, P$ and $n = 1, 2, \ldots, N$. To solve these problems, the passband region is discretized into the frequency points $\omega_i \in [0, \omega_p]$ for $i = 1, 2, \ldots, I$ and the range $-1 \leq \mu \leq 0$ into the points $\mu_j \in [-1, 0]$, $j = 1, 2, \ldots, J$. In many cases, $I = 10N$ and $J = 10$ are good selections to arrive at a very accurate solution. The resulting discrete minimization problem is to find $c_{pn}$ for $p = 1, 2, \ldots, P$ and $n = 1, 2, \ldots, N$ to minimize $\psi$ subject to the following condition:

$$\max_{\substack{1\leq i\leq I\\ 1\leq j\leq J}} |\tau_A(\Phi, \omega_i, \mu_j) - (N + \mu_j)| - \delta_p \leq 0 \quad (10)$$

and subject to the condition that the roots of $A(z, -1)$ are inside the unit circle. The above problems can be solved using a nonlinear optimization algorithm `fmincon` from the optimization toolbox provided by the MathWorks Inc.

### C. Optimization of Finite-Precision Filter

It has been experimentally proved that the parameter space defined above forms a space including the feasible space where the filter specifications are satisfied. After finding this larger space, all what is needed is to check whether in this space there exist combinations of the discrete coefficient values with which the given overall criteria are met.

This search can be carried out in a straightforward manner in the following steps:

*Step 1:* Find the sets of CSD numbers $C_{pn}$ for $p = 1, 2, \ldots, P$ and $n = 1, 2, \ldots, N$ between the smallest and largest values of each filter coefficient, i.e., for $p = 1, 2, \ldots, P$ and $n = 1, 2, \ldots, N$

$$\left\{C_{pn} \in \mathbb{CSD}_{(R,\, S_R)} \;\middle|\; c_{pn}^{(\min)} \leq C_{pn} \leq c_{pn}^{(\max)}\right\}, \quad (11)$$

where $c_{pn}^{(\min)}$'s and $c_{pn}^{(\max)}$'s, respectively, denote the smallest and largest values of the filter coefficients obtained using the infinite-precision optimization.

TABLE I
OPTIMIZED INFINITE-PRECISION PARAMETER VECTORS $\Phi^{(k)}$ FOR $k = 1, 2, \ldots, 8$ IN EXAMPLE 1

| | $\Phi^{(1)}$ | $\Phi^{(2)}$ | $\Phi^{(3)}$ | $\Phi^{(4)}$ |
|---|---|---|---|---|
| $c_{11}$ | −1.017 837 | −1.017 837 | −0.945 866 | −0.945 866 |
| $c_{21}$ | −0.072 051 | −0.072 051 | 0.029 743 | 0.029 743 |
| $c_{12}$ | 0.332 128 | 0.332 128 | 0.183 823 | 0.183 823 |
| $c_{22}$ | 0.318 005 | 0.318 005 | 0.159 433 | 0.159 433 |
| | $\Phi^{(5)}$ | $\Phi^{(6)}$ | $\Phi^{(7)}$ | $\Phi^{(8)}$ |
| $c_{11}$ | −0.840 753 | −0.840 753 | −1.000 069 | −1.000 069 |
| $c_{21}$ | 0.184 888 | 0.184 888 | −0.027 953 | −0.027 953 |
| $c_{12}$ | 0.290 113 | 0.290 113 | 0.362 039 | 0.362 039 |
| $c_{22}$ | 0.315 754 | 0.315 754 | 0.404 917 | 0.404 917 |

*Step 2:* Set $\mu = -1$ and based on the smallest and largest values of the filter coefficients, determine the smallest and largest values for the denominator coefficients $b_n^{(\min)}(\mu)$ and $b_n^{(\max)}(\mu)$ for $n = 1, 2, \ldots, N$, respectively.

*Step 3:* Quantize the smallest and largest values of the denominator coefficients determined at the previous step to the fixed number of bits as[2]

$$B_n^{(\min)} = q\left\lfloor \frac{b_n^{(\min)}(-1)}{q} \right\rfloor \quad \text{and} \quad B_n^{(\max)} = q\left\lceil \frac{b_n^{(\max)}(-1)}{q} \right\rceil \tag{12}$$

for $n = 1, 2, \ldots, N$, where $q = 2^{-S_R}$ with $S_R$ being the desired wordlength or the number of fractional bits. For each denominator coefficient, increase the value of the coefficient from $B_n^{(\min)}$ to $B_n^{(\max)}$ using a quantization step size equal to $q$. Then evaluate the phase delay response for each combination of discrete coefficient values to check whether the filter meets the specifications for this $\mu = -1$ case. Denote the solutions satisfying these specifications by $B_n^{(k)}$'s for $n = 1, 2, \ldots, N$ and for $k = 1, 2, \ldots, K$, where $K$ is the number of solutions satisfying the specifications.

*Step 4:* For all the solutions, find all the discrete coefficient value combinations in $C_{pn}$'s such that

$$B_n^{(k)} = \sum_{p=1}^{P} C_{pn}(-1)^p \tag{13}$$

for $n = 1, 2, \ldots, N$ and evaluate the phase delay responses for all the combinations of these values to check whether the filter meets the given specifications for $\mu_j \in [-1,\ 0]$ for $j = 1, 2, \ldots, J$.

*Step 5:* From the solutions satisfying the specifications at Step 4 choose the one with the minimum implementation cost as given by Eq. (9).

The above algorithm is based on the fact that for the fractional delay $\mu$ equal to $-1$, the denominator coefficients are given as signed sums of the filter parameters, that is, $b_n(-1) = -c_{1n} + c_{2n} - c_{3n} + \cdots$ for $n = 1, 2, \ldots, N$. By first finding the discrete values for the denominator coefficients at Step 3 in such a manner that $\tau_A(\Phi,\ \omega,\ -1)$ meets the criteria of Eq. (4) and, then, finding for the solutions satisfying these criteria the combinations of discrete coefficient values at Step 4 in such a way that conditions of Eq. (13) are met, makes the overall quantization scheme considerably faster.

## V. NUMERICAL EXAMPLES

### A. Example 1

It is required that $\Omega_p = [0,\ 0.75\pi]$ and $\delta_p \leq 0.05$. The given specifications are met by $N = 2$ and $P = 2$ [9]. For this infinite-precision filter the maximum phase delay error is 0.033 80.

[2]Here, $\lfloor x \rfloor$ stands for the largest integer that is smaller than or equal to $x$ and $\lceil x \rceil$ for the smallest integer larger than or equal to $x$.

TABLE II
THE SMALLEST AND LARGEST VALUES FOR THE DENOMINATOR COEFFICIENTS IN EXAMPLE 1

| $n$ | $b_n^{(\min)}(-1)$ | $b_n^{(\max)}(-1)$ | $B_n^{(\min)}$ | $B_n^{(\max)}$ |
|---|---|---|---|---|
| 1 | 0.945 79 | 1.025 64 | $31 \cdot 2^{-5}$ | $32 \cdot 2^{-5}$ |
| 2 | −0.024 39 | 0.042 88 | $0 \cdot 2^{-5}$ | $1 \cdot 2^{-5}$ |

TABLE III
THE DENOMINATOR COEFFICIENT VALUES $B_n^{(k)}$'S FOR THE SOLUTIONS SATISFYING THE SPECIFICATIONS IN EXAMPLE 1

| | | |
|---|---|---|
| $B_1^{(1)} = 31 \cdot 2^{-5}$ | $B_1^{(2)} = 31 \cdot 2^{-5}$ | $B_1^{(3)} = 32 \cdot 2^{-5}$ |
| $B_2^{(1)} = 0 \cdot 2^{-5}$ | $B_2^{(2)} = 1 \cdot 2^{-5}$ | $B_2^{(3)} = 1 \cdot 2^{-5}$ |

TABLE IV
PERMISSIBLE DISCRETE COEFFICIENT VALUE COMBINATIONS IN $C_{pn}$ FOR THE FIRST SOLUTION IN EXAMPLE 1

| $l$ | $C_{11}^{(l)}$ | $C_{21}^{(l)}$ | $l$ | $C_{12}^{(l)}$ | $C_{22}^{(l)}$ |
|---|---|---|---|---|---|
| 1 | $-1$ | $-2^{-5}$ | 1 | $2^{-2} - 2^{-4}$ | $2^{-2} - 2^{-4}$ |
| 2 | $-1 + 2^{-5}$ | 0 | 2 | $2^{-2} - 2^{-5}$ | $2^{-2} - 2^{-5}$ |
| 3 | $-1 + 2^{-4}$ | $2^{-5}$ | 3 | $2^{-2}$ | $2^{-2}$ |
| 4 | $-1 + 2^{-3}$ | $2^{-3} - 2^{-5}$ | 4 | $2^{-2} + 2^{-5}$ | $2^{-2} + 2^{-5}$ |
| | | | 5 | $2^{-2} + 2^{-4}$ | $2^{-2} + 2^{-4}$ |

The optimized parameter vectors $\Phi^{(k)}$ for $k = 1, 2, \ldots, 8$ after the infinite-precision optimization are shown in Table I. In this table, $\Phi^{(k)}$ for $k = 1, 2, 3, 4$ are the optimized parameter vectors for $\psi = c_{11}$, $\psi = c_{21}$, $\psi = c_{12}$, and $\psi = c_{22}$, respectively, and $\Phi^{(k)}$ for $k = 5, 6, 7, 8$ are solutions for $\psi = -c_{11}$, $\psi = -c_{21}$, $\psi = -c_{12}$, and $\psi = -c_{22}$, respectively. The corresponding smallest and largest values for the denominator coefficients are shown in Table II.

For this filter, $R$, the maximum number of powers-of-two terms, is two, whereas five fractional bits ($S_R = 5$) are required to meet the phase delay specifications.[3] The smallest and largest values of the denominator coefficients quantized to this wordlength are also shown in Table II. As can be seen from this table, there exist only two permissible discrete coefficient values for both of the denominator coefficients and, therefore, the number of function evaluations at Step 3 is four.

For this problem, there exist three finite-precision solutions satisfying the specifications at Step 3. The denominator coefficient values for these solutions are shown in Table III. For the first solution, the permissible discrete coefficient value combinations in $C_{pn}$ such that $B_1^{(1)} = C_{11}^{(l)}(-1)^1 + C_{21}^{(l)}(-1)^2$ for $l = 1, 2, 3, 4$ and $B_2^{(1)} = C_{12}^{(l)}(-1)^1 + C_{22}^{(l)}(-1)^2$ for $l = 1, 2, \ldots, 5$ are shown in Table IV. The phase delay specifications are met by selecting $c_{11} = C_{11}^{(2)} = -1 + 2^{-5}$, $c_{21} = C_{21}^{(2)} = 0$, $c_{12} = C_{12}^{(3)} = 2^{-2}$, and $c_{22} = C_{22}^{(3)} = 2^{-2}$. In this case, only one adder is needed to implement all the coefficients, whereas the number of structural adders is six [cf. Eq. (9)].

The direct evaluation of all the discrete coefficient combinations between the smallest and largest values of the filter coefficients results in 960 function evaluations. For the proposed algorithm, only four evaluations are needed at Step 3, whereas the number function evaluations required at Step 4 are 20, 16, and 16 for the solutions $A^{(k)}$ for $k = 1, 2, 3$, respectively, that is, the overall number of function evaluations is 56. In this case, there exists 16 finite-precision filters satisfying the specifications. The CPU time required when using a MATLAB 6.5 program

[3]In this case, four fractional bits is the shortest wordlength for which there exist discrete values between all the smallest and largest values of the denominator coefficients. However, for this coefficient wordlength, there is no solution satisfying the specifications.

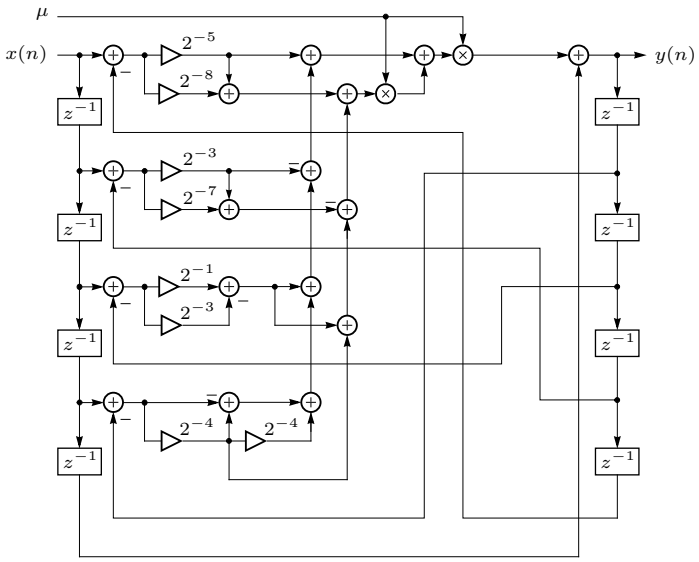


Fig. 1. Efficient implementation of the AFD all-pass filter in Example 2.

on a 500 MHz AlphaServer DS20 to evaluate all the combinations with $I = 40$ and $J = 10$ [cf. Eq. (10)] was 1.2 s.

### B. Example 2

The passband region is the same as in Example 1, whereas the maximum allowable phase delay error is divided by five, that is, $\delta_p \leq 0.01$. The given specifications are met by $N = 4$ and $P = 2$. For this infinite-precision filter the maximum phase delay error is 0.008 94.

For this filter, all the coefficient values can be represented as two or three powers-of-two terms, that is, $R$, the maximum number of powers-of-two terms, is three, whereas eight fractional bit ($S_R = 8$) are required to meet the phase delay specifications.

A total of only six adders and subtracters are required to implement all the multipliers for this coefficient representation form. The maximum phase delay error for the optimum solution is 0.009 91, whereas the radius of the outermost pole is 0.983 92 when $\mu = -1$. The CPU time required was 3.1 min. An implementation for this AFD all-pass filter requiring only 17 adders and subtracters is shown in Fig. 1. The implementation cost may be further reduced by using the matrix multiplier block as proposed by Dempster *et al.* in [11] or Macleod and Dempster in [12].

### C. Example 3

The passband region is the same as in Examples 1 and 2, whereas the maximum allowable phase delay error is $\delta_p \leq 0.005$. The given specifications are met by $N = 4$ and $P = 3$. For this infinite-precision filter the maximum phase delay error is 0.0040. In this case, the specifications are met by $R = 3$ and $S_R = 8$. The number of adders and subtracters required to implement all the filter coefficients is nine.

### D. Comparisons with Multiplierless AFD FIR Filters

Table V compares the performance and the complexity of the multiplierless AFD all-pass filter with those of the multiplierless AFD FIR filters implemented using the modified Farrow structure [3]. In this table, $\delta_a$ denotes the maximum deviation of the magnitude response from the unity on $\Omega_p$ for the AFD FIR filters, $r_{\max}$ denotes the radius of the outermost pole for the AFD all-pass filters when $\mu = -1$, whereas $N_{\mathrm{A}}$, $N_{\mathrm{M}}$, and $N_{\mathrm{D}}$ denote, respectively, the number of adders, multipliers,[4] and delays required for the overall implementation.

[4]The number of general multipliers required to implement the multiplication by $\mu^p$'s and $(1-2\mu)^p$'s for the AFD all-pass filters and AFD FIR filters implemented using the modified Farrow structure, respectively.

TABLE V
SUMMARY OF FILTER DESIGNS IN EXAMPLES 1, 2, AND 3

| Structure | $\delta_p$ | $\delta_a$ | $r_{\max}$ | $N$ | $P$ | $R$ | $S_R$ | $N_{\mathrm{A}}$ | $N_{\mathrm{M}}$ | $N_{\mathrm{D}}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| All-pass | 0.046 31 | 0 | 0.968 75 | 2 | 2 | 2 | 5 | 7 | 2 | 4 |
| All-pass | 0.009 91 | 0 | 0.983 92 | 4 | 2 | 3 | 8 | 17 | 2 | 8 |
| All-pass | 0.004 54 | 0 | 0.996 09 | 4 | 3 | 3 | 8 | 25 | 3 | 8 |
| Farrow | 0.008 69 | 0.009 00 | – | 12 | 3 | 3 | 7 | 38 | 3 | 36 |
| Farrow | 0.004 77 | 0.024 10 | – | 10 | 3 | 2 | 7 | 33 | 3 | 30 |

As can be seen from this table, the number of adders and delays is considerably smaller for the multiplierless AFD all-pass filters than for the AFD FIR filters even in the cases where the FIR filter is allowed to have a relatively large amplitude error. In this case, the complexity of the modified Farrow structure has been reduced by optimizing the filter coefficients in a proper manner as proposed by Yli-Kaakinen and Saramäki in [4], [5].

## ACKNOWLEDGMENT

This work was supported by the Academy of Finland, project No. 44876 (Finnish Centre of Excellence Program (2000–2005)). Juha Yli-Kaakinen was also financed by a postdoctoral research grants from the Academy of Finland, project Nos. 75492 and 105823.

## REFERENCES

[1] C. W. Farrow, "A continuously variable digital delay element," in *Proc. IEEE Int. Symp. Circuits Syst.*, Espoo, Finland, June 7–9 1988, pp. 2641–2645.

[2] T. I. Laakso, V. Välimäki, M. Karjalainen, and U. K. Laine, "Splitting the unit delay," *IEEE Signal Process. Mag.*, vol. 13, no. 1, pp. 30–60, Jan. 1996.

[3] J. Vesma and T. Saramäki, "Optimization and efficient implementation of FIR filters with adjustable fractional delay," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 1, Hong Kong, June 9–12 1997, pp. 2256–2259.

[4] J. Yli-Kaakinen and T. Saramäki, "Multiplier-free polynomial-based FIR filters with an adjustable fractional delay," in *Proc. IEEE Int. Conf. Electr. Circuits Syst. (ICECS 2002)*, vol. III, Dubrovnik, Croatia, Sept. 15–18 2002, pp. 1167–1170. [Online]. Available: http://alpha.cc.tut.fi/~ylikaaki/ICECS02.pdf

[5] ——, "Multiplier-free polynomial-based FIR filters with an adjustable fractional delay," submitted to *Circuits, Systems, and Signal Processing*, Special Issue on Computationally Efficient Digital Filters: Design Techniques and Applications, Nov. 2005.

[6] M. Makundi, V. Välimäki, and T. I. Laakso, "Closed-form design of tunable fractional-delay allpass filter structures," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 4, Sydney, Australia, May 6–9 2001, pp. 434–437.

[7] M. Makundi, T. I. Laakso, and V. Välimaki, "Efficient tunable IIR and allpass filter structures," *Electron. Lett.*, vol. 37, no. 6, pp. 344–345, Mar. 2001.

[8] C.-C. Tseng, "Design of 1-D and 2-D variable fractional delay allpass filters using weighted least-squares method," *IEEE Trans. Circuits Syst. I*, vol. 49, no. 10, pp. 1413 –1422, Oct. 2002.

[9] J. Yli-Kaakinen and T. Saramäki, "An algorithm for the optimization of adjustable fractional-delay all-pass filters," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. III, Vancouver, Canada, May 23–26 2004, pp. 153–156. [Online]. Available: http://alpha.cc.tut.fi/~ylikaaki/ISCAS04FD.pdf

[10] S. K. Mitra, *Digital Signal Processing: A Computer-Based Approach*. New York: McGraw-Hill, 1998.

[11] A. G. Dempster, O. Gustafsson, and J. O. Coleman, "Towards an algorithm for matrix multiplier blocks," in *Proc. Euro. Conf. Circuit Theory Design*, vol. III, Kraków, Poland, Sept. 1–4 2003, pp. 9–12.

[12] M. D. Macleod and A. G. Dempster, "Common subexpression elimination algorithm for low-cost multiplierless implementation of matrix multipliers," *Electron. Lett.*, vol. 40, no. 11, May 2004.