CORDIC REALIZATION OF THE TRANSVERSAL ADAPTIVE FILTER USING A TRIGONOMETRIC LMS ALGORITHM

Mrityunjoy Chakraborty, Anindya S. Dhar, Suraiya Pervin
Dept. of Electronics and Electrical Communication Engineering,
Indian Institute of Technology, Kharagpur-721302, W.B., India.
e-mail: {mrityun,asd,pervin}@ece.iitkgp.ernet.in.

ABSTRACT

This paper presents a class of pipelined CORDIC architectures for the LMS-based transversal adaptive filter. For this, an alternate formulation of the LMS algorithm is considered, obtained by expressing the mean square error as a convex function of a set of angle variables that are monotonically related to the filter tap weights. The proposed architectures employ microlevel pipelining and are adjustable to strike tradeoffs between throughput efficiency vis-à-vis hardware complexity.

1. INTRODUCTION

Since the last two decades, the implementation and application of CORDIC arithmetic [1] continue to evolve into very useful areas because of its numerical stability, efficiency in evaluating trigonometric functions and hyperbolic transformations, hardware compactness and computational simplicity [2]. In the field of signal processing, CORDIC method has been employed successfully for a wide range of applications like performing FFT, DCT, DST, SVD and other matrix operations, filtering and array processing [2]. For the case of the LMS-based adaptive filters, however, the CORDIC based approach has so far remained confined largely to lattice filters ([3]-[4]) and seemingly has not been extended to the transversal form, as, in the case of the former, the computations in each stage can be related easily to a set of hyperbolic operations, while no such direct hyperbolic, or, trigonometric interpretation exists for the computations present in the latter. In this paper, we first propose an alternate formulation of the LMS algorithm using a set of trigonometric variables which are monotonically related to the transversal filter coefficients. Subsequently, we present two CORDIC based architectures which implement the proposed trigonometric LMS (TLMS) algorithm. Simulation results highlighting the convergence behaviour of the TLMS algorithm are also presented.

2. A TRIGONOMETRIC FORMULATION OF THE LMS ALGORITHM

We begin by first considering the steepest descent search procedure that arises in the optimal FIR filtering problem. Given an input sequence $x(n)$, desired response $d(n)$ and a N-tap filter coefficient vector $\mathbf{w}(n) = [w_0, w_1, \ldots, w_{N-1}]^t$, the optimal filter $\hat{\mathbf{w}} = [\hat{w}_0, \hat{w}_1, \ldots, \hat{w}_{N-1}]^t$ is obtained by minimizing the mean square error (MSE) function $\varepsilon^2 = E[e^2(n)]$, where $e(n)$ is the error signal at the filter output and is given by $e(n) = d(n) - \mathbf{w}^t(n)\mathbf{x}(n)$, with $\mathbf{x}(n) = [x(n), x(n-1), \cdots, x(n-N+1)]^t$. The MSE ε^2 is a convex function of the filter coefficients w_k's, $k = 0, 1, \ldots, N-1$ and defines the so called error performance surface in an N+1 dimensional space. In the proposed alternative, we select a set of N positive numbers A_k's, $k = 0, 1, \ldots, N-1$, so that the minima of the error performance surface is contained in the hypercube with vertices: $[\pm A_0, \pm A_1, \ldots, \pm A_{N-1}]^t$ (In practice A_k's are taken to be some powers of 2 for convenience in hardware realization). Each tap weight w_k in the above range can then be expressed as $w_k = A_k \sin\theta_k$, $-\frac{\pi}{2} < \theta_k < +\frac{\pi}{2}$. Since each $w_k \in [-A_k, +A_k]$ maps uniquely to a $\theta_k \in [-\frac{\pi}{2}, +\frac{\pi}{2}]$, the MSE ε^2, when expressed as a function of θ_k's has a unique minima at $\hat{\theta}_k = \sin^{-1}(\hat{w}_k)$, $k = 0, 1, \ldots, N-1$, located within a hypercube in the θ space with vertices: $[\pm\frac{\pi}{2}, \pm\frac{\pi}{2}, \cdots, \pm\frac{\pi}{2}]$. Further, the function $\sin\theta_k$ is a monotonically increasing, continuous function of θ_k, as θ_k varies from $-\frac{\pi}{2}$ to $+\frac{\pi}{2}$, meaning

that, $\frac{\partial \varepsilon^2}{\partial \theta_k}$ has the same sign as that of $\frac{\partial \varepsilon^2}{\partial w_k}$ everywhere within the hypercube. In other words, the MSE does not exhibit any local minima within the specified hypercube.

In the proposed scheme, a steepest descent search is taken up in the θ space in order to reach $\hat{\theta}$. The gradient $\nabla_\theta \varepsilon^2$ is easily seen to be given by $\nabla_\theta \varepsilon^2 = -2\,\Delta(\mathbf{p} - \mathbf{Rw})$, where $\mathbf{w} = [A_0 \sin\theta_0, A_1 \sin\theta_1, \ldots, A_{N-1} \sin\theta_{N-1}(n)]^t$, $\mathbf{p} = E[\mathbf{x}(n)d(n)]$, $\mathbf{R} = E[\mathbf{x}(n)\mathbf{x}^t(n)]$ and Δ is a diagonal matrix with *j*-th diagonal entry given by $\Delta_{j.j} = A_j \cos\theta_j$, j = 0, 1, …, *N*-1. The iterate $\theta(i)$ arising in the *i*-th step of iteration is then updated as :

$$\theta(i+1) = \theta(i) - \mu \nabla_\theta \varepsilon^2 \,\big|_{\theta = \theta(i)}$$

where μ is some appropriate step size. To move from the steepest descent to the LMS form, we simply replace **R** and **p** by $\mathbf{x}(n)\mathbf{x}^t(n)$ and $\mathbf{x}(n)d(n)$ respectively in the expression for $\nabla_\theta \varepsilon^2$ in order to obtain an estimate of the gradient at index *n*. This leads to the so called "Trigonometric LMS (TLMS)" algorithm as follows:

$$\theta(n+1) = \theta(n) + \mu\, \Delta(n)\mathbf{x}(n)e(n), \qquad (1)$$

$$e(n) = d(n) - \sum_{k=0}^{N-1} A_k \sin\theta_k(n)x(n-k) \qquad (2)$$

The TLMS algorithm is particularly suitable for CORDIC based realization, since the two quantities: $A_k \sin\theta_k(n)x(n-k)$ and $A_k \cos\theta_k(n)x(n-k)$, $k = 0, 1, \ldots, N-1$, required for filtering by and updatation of the *k*-th coefficient respectively *can be computed simultaneously by engaging only one CORDIC processor*. For pipelined realization, it may, however, be more appropriate to consider the trigonometric analog of an approximate version of the LMS algorithm, popularly known as the "Delayed LMS"(DLMS) algorithm [5], where the filter coefficients at the *n*-th index are updated using a past estimate of the gradient, say, for index $(n-L)$, where *L* is an integer. The correction term in the weight update formula then gets modified to $\mu\mathbf{x}(n-L)e(n-L)$ and the resulting *L* cycle delay in the error feedback path is used for retiming purpose. The trigonometric analog of the DLMS algorithm can be easily worked out by substituting **R**, **p** and **w** in the gradient expression by $\mathbf{x}(n-L)\mathbf{x}^t(n-L)$, $\mathbf{x}(n-L)d(n-L)$ and $[A_0 \sin\theta_0(n-L), A_1 \sin\theta_1(n-L), \ldots, A_{N-1} \sin\theta_{N-1}(n-L)]^t$ respectively and is given by

$$\theta(n+1) = \theta(n) + \mu\Delta(n-L)\mathbf{x}(n-L)e(n-L) \qquad (3)$$

It may, however, be noted that unlike the conventional LMS, it is very difficult to prove convergence of the TLMS and the delayed TLMS (DTLMS) algorithms analytically owing to the presence of nonlinearities in the form of trigonometrical quantities in (1), (2) and (3). Both the TLMS and the DTLMS algorithms, however, have been simulated extensively in the context of a wide class of applications and promising convergence results observed in each case. In this paper, we present simulation results for equalizing an AWGN channel with transfer function $H(z) = (1 + 2z^{-1})(1 - \frac{1}{2}z^{-1})$ and noise variance .077. The transmitted symbols were chosen from an alphabet of 16 equispaced, equiprobable discrete amplitude levels with transmitted signal power of 10 dB. A 9 tap equalizer with centre placed at the 5-th tap position was used for equalizing the channel and a step size of μ = .0004 was adopted for weight updatation by the DTLMS algorithm. The resulting output error characteristics, displayed in Fig. 1 by plotting $e(n)$ vs. *n*, confirms satisfactory convergence properties of the proposed method.

3. PROPOSED CORDIC ARCHITECTURE

The CORDIC algorithm [1] provides an efficient way of implementing (2) and (3). This algorithm essentially rotates a two dimensional vector by running the iterations: $x_{i+1} = x_i - \delta_i 2^{-i} y_i$, $y_{i+1} = \delta_i 2^{-i} x_i + y_i$ and $\varepsilon_{i+1} = \varepsilon_i - \delta_i \arctan(2^{-i})$, where $\delta_i = \mathrm{sgn}(\varepsilon_i)$, *i*=0, 1,…, *M*-1, *M* being the wordlength of implementation. After *M* iterations, for large *M*, $x_M \to k(x_0 \cos\theta - y_0 \sin\theta)$, $y_M \to k(x_0 \sin\theta + y_0 \cos\theta)$ and $\varepsilon_M \to 0$ where $k = 1/\prod_{i=0}^{M-1} \cos(\arctan(2^{-i}))$ is the so called scale factor having a constant value for a particular machine with wordlength *M* and (x_0, y_0) is the initial two dimensional vector, $\varepsilon_0 = \theta$ being the desired angle of rotation. Fig. 2 shows a CORDIC realization of a *N* tap TLMS-based adaptive filter which achieves microlevel pipelining by using pipelined CORDIC processor units. Note that each stage within the CORDIC processor needs to employ only adder/subtractors, while the shifting operations implicit in the multiplications by 2^{-i} can be carried out simply by adopting fixed oblique bus connection instead of hardware-hungry variable shifters. Since the critical path delay arising from the CORDIC processors as well as from the pipelined multipliers amounts to that of a single adder/subtractor, this architecture can indeed process very high throughput data, typically of the order of hundreds of megahertz.

It is, however, possible to achieve considerable reduction in hardware complexity, as well as in latency if the architecture is allowed to operate at a clock faster than the input. This will involve replacing group of CORDIC blocks (also the pipelined multipliers) by single units and adopting appropriate sequencing of the data through these units. As a special case, we consider

the architecture shown in Fig. 3 which employed a single pipelined CORDIC processor and a single pipelined multiplier driven by a primary clock that is N+1 times faster than the input. Three FIFOs, R1, R2 and R3 are used for proper sequencing of the input data, tap weights, and intermediate results required for weight updating. N number of samples are circulated through R1 at the primary clock rate, while a new sample is introduced at every N+1-th clock cycle by flipping a MUX. The CORDIC pipeline computes $A_k \sin\theta_k(n)x(n-k)$ and $A_k \cos\theta_k(n)x(n-k)$ concurrently. However, as it takes N+1 additional clock cycles for the hardware to compute the error term $e(n)$ in (2), the latter terms are buffered in R2 of length N+1 to maintain the synchronism required between the indices of $e(.)$ and $\Delta(.)\mathbf{x}(.)$ while evaluating (3). Note that after every N clock cycles, a *don't care state* is introduced into the CORDIC pipe to balance one extra clock cycle (N+1-th cycle for a N tap filter) required for computing (2). The FIFO R3, having length N+1, is used to store the angles θ_k and the position of the tapping from R3 is adjusted to maintain correctness of the input of the CORDIC unit vis-à-vis the data input. Finally, the CORDIC pipeline introduces a scale factor (≈ 1.6) implying that the stepsize in (3) should be chosen as $\mu / 2.56$.

4. DISCUSSION AND CONCLUSION

In this paper, we have presented an alternate formulation of the LMS algorithm by mapping the filter tap weights to a set of trigonometric variables. The resulting algorithm directly conforms to CORDIC based realizations and two architectures have been presented both of which employ microlevel pipelining by engaging pipelined CORDIC blocks. The proposed algorithm has also been shown to process satisfactory convergence characteristics through extensive simulation studies.

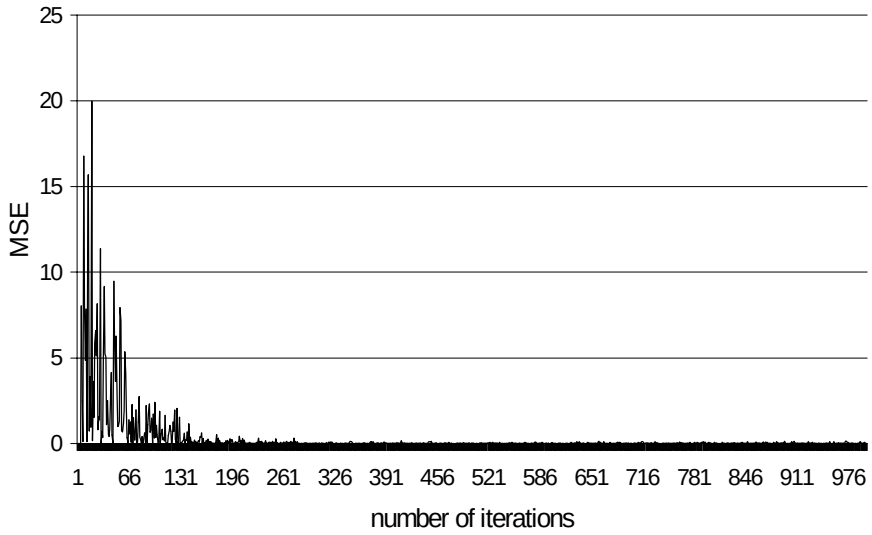

Fig. 1. Convergence behaviour of the delayed TLMS algorithm.

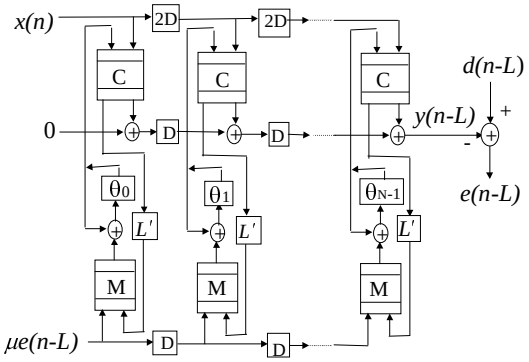

Fig. 2. Transversal adaptive filter architecture using N number of pipelined CORDIC blocks(C) and pipelined multipliers (M).

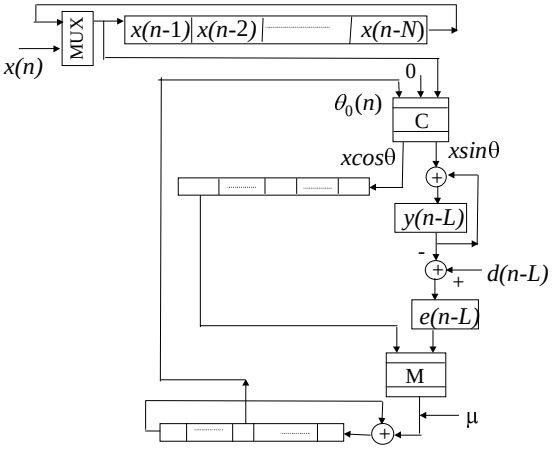

Fig. 3. Transversal adaptive filter architecture using one pipelined CORDIC block and one pipelined multiplier.

5. REFERENCES

[1] J. E. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. Electron. Comput.*, vol. EC-8, no. 3, pp. 330-334, Sept. 1959.

[2] Y. H. Hu, "CORDIC-Based VLSI architecture for digital signal processing", *IEEE Signal Processing Magazine*, vol. 9, no 3, pp 16-3, July 1992..

[3] Y. H. Hu and H. E. Liao, "CALF: a CORDIC adaptive lattice filter", *IEEE Trans. Signal Processing*", vol. 40, no. 4, pp. 990-993, April 1992.

[4] Yu Hen Hu, "On the convergence of the CORDIC adaptive lattice filtering (CALF) algorithm", *IEEE Trans. Signal Processing*, vol. 46, no. 7, pp. 1861-1871, July 1998.

[5] G.Long, F. Ling, J. G. Proakis, "The LMS algorithm with delayed coefficient adaptation", *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-37, pp. 1397-1405, September 1989.

[6] M. Chakraborty, A. S. Dhar and S. Pervin, "CORDIC realization of transversal adaptive filters using a Trigonometric/Hyperbolic LMS algorithm", communicated to *IEEE Trans. Circuits and Systems, part II* .

******