Active Noise Control using Variable step-size Griffiths' LMS (VGLMS) algorithm on Real-Time platform

V. Vinay, V. Roopashree, S.V. Narasimhan, M.Shivamurti
Aerospace Electronics & Systems Division
National Aerospace Laboratories, Bangalore-560017
Council of Scientific and Industrial Research (CSIR), New Delhi, INDIA
narasim@nal.res.in

Abstract—**This paper proposes implementation of Griffith's Variable step-size algorithm for Active Noise Control (ANC) on ADSP-TS201 EZ-Kit Lite. The dual computational units and execution of up to four instructions per cycle which are special features over other processors are best utilized to generate an optimized code. The VGLMS provides improved secondary path estimation and computations involved are marginal as the same gradient is used for step-size computation and coefficient adaptation. The improved secondary path estimate, in turn improves the ANC performance. Further, variable step-size algorithm is used for the main-path to achieve faster convergence. Both for narrowband (fundamental and its harmonics) and broadband noise fields, for a duct the attenuation achieved is 25 dB and 15 dB respectively. The program execution time was only 1.25% for an input sampling rate of 1 KHz which indicates the utility of the special features of the processor considered. Further these features have enabled in bringing down the program memory requirement in the implementation of the algorithm.**

Keywords- Active Noise Control, Filtered-X LMS (FXLMS) Algorithm, Variable Step-Size Griffiths' Algorithm, ADSP-TS201 EZ-Kit Lite

I. INTRODUCTION

The LMS adaptive algorithm due to its simplicity is widely used for many applications like system identification, channel equalization, echo cancellation, noise removal, adaptive coding/compression and active noise control. Its structure and operation are well-suited for implementation on Digital Signal Processor (DSP) chips, due to the algorithm's extensive use of the multiply / accumulate (MAC) operations. In ANC, to account for the passage of the error through secondary-path (SP), for the LMS, original input(X) filtered by SP is used. To ensure good performance of FXLMS algorithm, the SP estimate has to be accurate. In FXLMS algorithm, larger step-size results in faster convergence but with high convergence error, whereas smaller step size results in low convergence error but with slower convergence. To achieve both fast convergence and low convergence error variable step size algorithm is used, where larger step-size is used prior to convergence and the step size is reduced near convergence to get a smaller convergence error. Further, in ANC using FXLMS a high observation noise affects secondary path identification. To overcome this, Griffiths' algorithm which takes care of the effect of observation noise on the gradient is used.

Active Noise Control (ANC) is sound field cancellation by electro-acoustical means. The antinoise built by adaptive filter is sent out by the loud speaker and this interacts with the existing (primary) noise field over a physical space resulting in a residual noise field. This is realized by detecting the noise, and processing it via a suitable electronic controller (digital signal processor) using an adaptive algorithm. The advent of cost-effective digital signal processing hardware has been the principal enabler of practical implementation of ANC and has driven intensive interest in this field in the last decade. Hence many of the signal processing tasks that were conventionally performed by analog means are realized today by less expensive and often more reliable digital hardware. Indeed, for many signals with extremely wide bandwidths, analog or optical signal processing is the possible solution; but in cases where digital circuits are available and have sufficient speed to perform signal processing, they are preferred. ANC is an efficient approach for reducing low frequency noise in the range of 50Hz to 500Hz. Hence, a sampling rate of 1 KHz is required, which means that an interval of 1ms is available for the computation of antinoise. The 1ms time constraint and algorithm's computational complexity calls for the use of digital signal processors and programming practice with good optimization.

This paper is organized as follows. Section II presents an introduction to active noise control algorithms. The computational load in FXLMS (FXLMS) and VGLMS for a single channel ANC system is compared in section III. Section IV presents the features of processor architecture and all key concepts used for efficient assembly level programming to produce an optimized code. Real-time implementation of ANC is explained in section V. Results of simulations in section VI shows that VGLMS adaptive algorithm gives better performance compared to FXLMS.

II. ACTIVE NOISE CONTROL ALGORITHMS

A. FXLMS Implementation

A schematic of single channel active noise control system is shown in Fig. 1. The antinoise $y(n)$ in (1a), built by adaptive filter $W(k)$ in (1c) is given out by the loud speaker and this interacts with the existing (primary) noise field over a physical space resulting in a residual noise field. Effectively, the error $e(n)$ in (1b) is available only after it passes through an acoustic path $S(k)$, the secondary-path (SP).

$$y(n) = \sum_{k=0}^{L-1} W_k(n)\, x(n-k) \tag{1a}$$

$$e(n) = e_d(n) + e_v(n) = [d(n) - y(n)] * s(n) + v(n) * s(n) \tag{1b}$$

where ‘ $*$ ’ indicates convolution operation. $d(n)$ is the undesired noise to be cancelled. The components $e_d(n)$ and $e_v(n)$ are due to residual noise field and SP excitation $v(n)$ respectively. Since during adaptation, $y(n)$ is changing, $e(n)$ becomes nonstationary. In ANC, $W(k)$ is adapted using FXLMS algorithm given by (1c).

$$W_k(n+1) = W_k(n) + \mu\, e(n)\, x'(n-k), \quad k = 0.1,\ldots,L-1 \tag{1c}$$

where, $$x'(n) = S_k(n) * x(n) \tag{1d}$$

As $S(k)$ is not known, its estimate $\hat{S}(k)$ has to be found using

$$\hat{S}_k(n+1) = \hat{S}_k(n) + \mu_s\, Z_e(n)\, v(n-k) \quad k = 0,1,\ldots,M-1 \tag{1e}$$

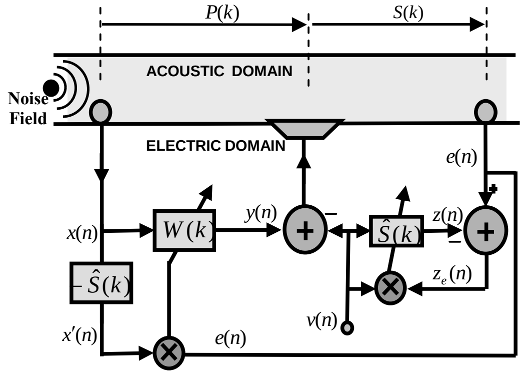

Fig.1 Block diagram of Single Channel ANC system

The performance of ANC depends on the accuracy of SP estimate. The estimation of $\hat{S}(k)$ is done on online basis using white noise $v(n)$ due to its merits. $\hat{S}(k)$ is updated by NLMS algorithm using the step-size μ. The SP identification error is,

$$z_e(n) = e(n) - z(n) = e_d(n) + e_v(n) - v(n) * \hat{s}(n) \tag{1f}$$

The identification has to be done in the presence of a strong observation noise $e_d(n)$. The level of $v(n)$ has to be below the primary noise level (by about 25dB) as it passes through the loudspeaker and decides the noise floor level or the amount of attenuation achievable by ANC. Further, the identification should be fast as it is required in advance for filtering and generating the reference signal, calling for a larger adaptation step-size which results in a higher mis-adjustment. All these factors contribute to phase error and delay in the SP estimate, deteriorating the performance of ANC. Therefore, an accurate SP estimate is absolutely necessary for improving the performance of feed forward ANC.

B. VGLMS Implementation

The implementation of a robust algorithm in ANC which uses VGLMS for SP estimate and Variable step-size LMS for Main path (MP) estimate is described here. The VGLMS algorithm is used for improving the SP identification for ANC [1]. $G_k(n)$ given by (2a) of this algorithm enables better estimate of $S(k)$ even in the presence of a strong primary noise field, as its gradient and the step-size both have good noise immunity. $G_k(n)$ also enables faster adaptation, as only its noisy part is averaged. The choice of ‘ σ ’ in (2c) facilitates to effectively handle the nonstationary observation noise scenario of ANC. In this implementation, it is clear that computational load is increased due to calculation of cross-correlation term. But this will be shadowed by the improvement in convergence. This additional computational load is marginal and is well within the critical sampling time of 1ms. Equations (2a) to (2f) represent the extra operations performed in VGLMS algorithm which increase the computational load.

$$G_k(n+1) = \beta_g\, G_k(n) + (1-\beta_g)\, e(n)\, v(n-k) \tag{2a}$$

$$\hat{S}_k(n+1) = \hat{S}_k(n) + \mu(n)\,[G_k(n) - z(n)\, v(n-k)] \tag{2b}$$

$$\mu_s(n+1) = \alpha\, \mu_s(n) + \sigma \sum_{k=0}^{M-1} G_k^2(n) \tag{2c}$$

Here, in addition to applying VGLMS to secondary path identification, variable step-size LMS algorithm is applied to main path adaptation**.** This further increases the performance with respect to convergence speed and convergence error when compared to FXLMS based ANC system.

$$G_k(n+1) = \beta_{VS}\, G_k(n) + (1-\beta_{VS})\, e(n)\, x'(n-k) \tag{2d}$$

$$W_k(n+1) = W_k(n) + \mu(n)\, e(n)\, x'(n-k) \tag{2e}$$

$$\mu(n+1) = \alpha\, \mu(n) + \sigma \sum_{k=0}^{L-1} G_k^2(n) \tag{2f}$$

III. COMPUTATIONAL CONSIDERATION

In this section, the computational requirements of FXLMS and VGLMS algorithms are compared. Single channel ANC with FXLMS adaptive algorithm requires 2L+4M+1 multiplications and 2L+4M-2 additions. Table-I gives details of the additional computations in ANC with VGLMS adaptive algorithm compared to ANC with FXLMS algorithm given by (2a) to (2f).

TABLE I. ADDITIONAL COMPUTATIONS IN VGLMS

VGLMS algorithm computations		
Equation	**Multiplications**	**Additions**
Cross-correlation (2a)	2M+1	M
Secondary path estimate (2b)	2M	2M
Step size adaptation (2c)	M	M
Cross-correlation (2d)	2L+1	L
Main path estimate (2e)	L	L
Step size adaptation (2f)	L	L
Total	**4L+5M+2**	**3L+4M**

where 'L' refers to the main-path length L=128 and 'M' refers to the secondary-path length M=64.

IV. ADSP-TS201 PROCESSOR DETAILS

The key features of the processor used for implementation are listed below.

A. Processor Architecture

The ADSP-TS201 is a 128-bit, high performance TigerSHARC processor [2, 3, 4]. The processor core operates at a frequency of 500MHz. The processor's architectural features like dual computation units and dual integer ALUs are utilized to the fullest here. The processing core of the ADSP-TS201 processor reaches exceptionally high performance using the features — computation pipeline, dual computation units, execution of up to four instructions per cycle and access of up to eight words per cycle from memory. The TigerSHARC processor has two general purpose 64-bit timers which are free running down counters and produce a timer interrupt when count reaches zero. These timers are set appropriately to generate interrupt every 1ms. The EZ-Kit Lite has onboard 24-bit analog-to-digital convertor (ADC) AD-1871 and 24-bit digital-to-analog convertor (DAC) AD-1854 [5, 6]. These operate at a rate of 96 KHz, which is higher than the required sampling rate. Hence, timer interrupt is used for sampling.

B. Key Concepts Used For Efficient Implementation

1) Circular Buffer Addressing

Any digital signal processor is designed with limited memory resources, and adding external memory can turn out to be expensive and also take a toll on speed of operation. In this situation, processing real-time data, which is bound to be large, is impossible. This calls for the use of circular buffering, where only recent samples of certain length are used for processing. Here, a circular buffer of length 'L' is created for input data by using the circular buffer addressing mode provided by ADSP-TS201 processor.

2) Dual Computation Units

The ADSP-TS201 processor core contains two compute blocks — compute block X and compute block Y. Each compute block has a multiplier and an arithmetic logic unit (ALU). These are used to perform two MAC operations in each cycle. This is achieved by using single instruction multiple data (SIMD) instructions like

$$FR16 = R0 * R1;;$$

$$FR17 = R2 * R3; FR20 = R20 + R16;;$$
$$FR18 = R4 * R5; FR20 = R20 + R17;;$$

In the above example, computation pipelining is achieved by operating on the results of operation from previous cycle to make up for one cycle delay.

3) Data Alignment Buffer (DAB)

Processor implementation reaches high efficiency when eight words are loaded from memory to registers in one cycle. This can be done using quad-word access. For optimized implementation, a quad-word from two different buffers is loaded into 'X' block and 'Y' block respectively in one cycle. Whenever quad-word accesses have to be made from a circular buffer, DAB register must be used. DAB is a single 128-bit FIFO register which takes non-aligned data as input and gives out aligned data. The use of DAB ensures that the data is quad-aligned, which is a requirement for quad-word access. Further, by using both integer ALUs for addressing, up to eight words can be loaded into registers from memory in a single instruction cycle; thus increasing the efficiency of implementation. For example,

$$yr3:0 = ydab\ q[j0+=4];;$$
$$xr3:0 = xdab\ q[k0+=4];;$$

Both instructions occupy a single instruction line, in the above example. Hence, they are executed in the same instruction cycle.

V. REAL TIME IMPLEMENTATION OF ANC ALGORITHM

Even though the input data required for real time implementation is very large (system may have to operate continuously for a day), the processor cannot handle this quantity due to limited memory resources. Hence, it is desirable to retain only the most recent samples specified by the length of the filter (L or M). This necessitates for the use of circular buffers for efficient implementation, where only the pointer to the current sample is shifted and the contents of that location are replaced by the newest sample. In ANC, to operate (say filtering) on input, the data should be loaded from memory to registers of the processor. This delay in load operation can be minimized by quad-word access. To do this, data must be quad-aligned. Since input data is stored circularly, data may not be quad-aligned. Hence to facilitate quad-word load to registers, DAB access must be used. This will ensure that the data loaded will be quad-aligned. As mentioned earlier, FXLMS algorithm has extensive use of MAC operations. Hence, the speed of execution of ANC algorithm can be increased by increasing the number of MAC operations per cycle. Using dual computation units and SIMD instructions appropriately, two MAC operations can be performed in one cycle.

VI. Simulation Results

In this section, the simulation results of FXLMS and VGLMS algorithms for a single channel ANC system on ADSP-TS201 processor are presented. The parameter values set in FXLMS and VGLMS adaptive algorithms are listed below in Table-II and Table-III respectively. Table-IV shows the execution time of FXLMS and VGLMS algorithms on ADSP-TS201 processor. For simulation, internally generated white noise is taken as reference input signal.

TABLE II. Parameters for FXLMS

Parameter	**μ**	**μ_s**	**β**	**β_s**
Value	0.01	0.01	0.999	0.999

TABLE III. Parameters for VGLMS

Parameter	**μmax**	**μmin**	**$\mu_s$max**	**$\mu_s$min**	**β_g**
Value	1	0.02	1	0.005	0.99
Parameter	**β**	**β_s**	**α**	**σ**	**β_{vs}**
Value	0.999	0.999	0.999	0.001	0.99

TABLE IV. Execution Time on ADSP-TS201 Processor

Algorithm Name	**Execution time**
FXLMS	8.5 µs
VGLMS	12.5 µs

From Fig. 2, the error signal converges faster in VGLMS than FXLMS. Variable step-size algorithm applied to main-path and secondary-path adaptation increases the speed of convergence and decreases the convergence error.

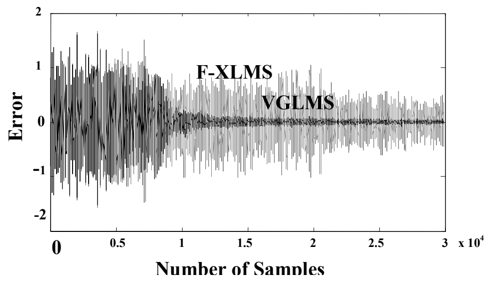

Fig. 2 Convergence Error Plot for FXLMS and VGLMS

Fig. 3 shows a plot of $\Delta_m(n) = \sum_{k=1}^{L}[P_k(n) - W_k(n)]^2$ for main path and Fig. 4 show a plot of $\Delta_s(n) = \sum_{k=1}^{M}[S_k(n) - \hat{S}_k(n)]^2$ for secondary path where P_k and S_k are the actual weight vectors of main path and secondary paths used for simulation respectively. It is clear from these figures that there is a mis-alignment in actual and adapted weight vectors in FXLMS, which results in higher convergence error.

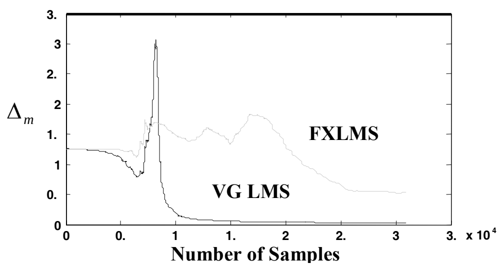

Fig. 3 Main-path Δ_m

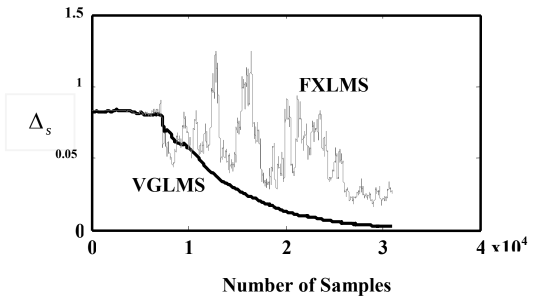

Fig. 4 Secondary-path Δ_s

VII. Conclusion

This paper shows successful implementation of VGLMS based Single Channel ANC system on Analog Devices' ADSP-TS201 processor. Performance of VGLMS is compared with FXLMS algorithm, and the results prove that VGLMS gives improvement in convergence speed and convergence error.

Acknowledgment

We would sincerely like to acknowledge and extend our heartfelt gratitude to Mrs. Veena.S and Mr. Lokesha.H, for their motivation and support in understanding of the DSP algorithm.

References

[1] S.V.Narasimhan, S.Veena, Lokesha.H, "Variable step-size Griffiths' algorithm for improved performance of Feedforward/Feedback Active Noise Control", Signal, Image and Video processing, Springer link, 2004.

[2] ADSP-TS201S EZ-KIT Lite Evaluation System Manual, Revision 3.1, April 2007, Analog Devices Inc.

[3] ADSP-TS201 TigerSHARC Processor Programming Reference, Revision 1.1, April 2005, Analog Devices Inc.

[4] ADSP-TS201 TigerSHARC Processor Hardware Reference, Revision 1.1, December 2004, Analog Devices Inc.

[5] Analog Devices' Stereo Audio, 24-bit, 96KHz, Multibit AD-1871 $\sum - \nabla$ ADC.

[6] Analog Devices' Stereo Audio, 24-bit, 96KHz, Multibit AD-1854 $\sum - \nabla$ DAC.

[7] www.analog.com/processors