Buletinul Ştiinţific al Universităţii "Politehnica" din Timişoara

Seria ELECTRONICĂ şi TELECOMUNICAŢII
TRANSACTIONS on ELECTRONICS and COMMUNICATIONS

Tom 49(63), Fascicola 2, 2004

Variable Step Size Affine Projection Adaptive Algorithm Implementation

Sorin Zoican [1]

Abstract - The paper presents a new variable step-size adaptive algorithm for affine projection (VSS-AP) with a faster convergence and lower misadjustment. The VSS-AP algorithm is compared with the LMS algorithm. The possibility of real time implementation of this algorithm is investigated, using a DSP microcomputer (ADSP 21161 – Analog Devices).

Keywords: variable step-size adaptive algorithm, digital signal processor (DSP)

I. INTRODUCTION

The performance of LMS or NLMS algorithms are drastically deteriorate if the input signal is colored input, not a white noise one. Many techniques were proposed in the literature in order to overcome this. The main idea in all of these techniques is to control the adaptation parameter (the step size) but the performance (in terms of convergence speed and final residual error) depends of the estimation of how far the filter is from optimal performance.

If this estimation take into consideration only the input vector data (that is the current sample and some delayed input sample) the performance will be not very good, of course depends of the input signal characteristics. On the other hand if the estimation of the filter performance uses an input matrix the performance will be better even the input signal is colored.

In the second case the computational effort will be significantly increased and the round off errors can compromise the algorithm convergence.

II. THE VARIABLE STEP-SIZE ALGORITHM

The VSS-AP algorithm take into consideration K input vectors.

We note: d(i) – desired signal, $\mathbf{w}^0$ – the coefficients of the finite impulse response (FIR) plant filter of length M, $\mathbf{u}_i$ – the (1 x M) input vector, $\mathbf{U}_i = [\mathbf{u}_i \ \mathbf{u}_{i-1} \ \dots \ \mathbf{u}_{i-K+1}]^T$ – the (K x M) input matrix (T denote transpose), $\mathbf{d}_i = [\,d(i)\ d(i-1) \ \dots \ d(i-K+1)]^T$ – a K x 1 vector , $\mathbf{e}_i$ – the residual error (K x 1) vector.

The VSS-AP algorithm is the following:

$$d(i) = \mathbf{u}_i \mathbf{w}^0 \qquad (1)$$

$$\mathbf{w}_i = \mathbf{w}_{i-1} + \mu(i)\, \mathbf{U}^*_i (\mathbf{U}^*_i \mathbf{U}_i)^{-1} \mathbf{e}_i \qquad (2)$$

$$\mathbf{e}_i = \mathbf{d}_i - \mathbf{U}_i \mathbf{w}_{i-1} \qquad (3)$$

The criterion used to control the algorithm performance is the closing to the optimal performance in terms of minimum mean square deviation (MED) given by $E\{\|\mathbf{w}^0 - \mathbf{w}_i\|^2\}$, where $E\{.\}$ represents the expectation operator.

Let $\mathbf{p}_i = \mathbf{U}^*_i (\mathbf{U}^*_i \mathbf{U}_i)^{-1} \mathbf{U}_i \mathbf{w}_{i-1}$ the affine projection of $\mathbf{w}_{i-1}$ onto the range space of matrix $\mathbf{U}^*_i$

The minimum mean square deviation ca be written using the expression (2) and by minimizing the MED with respect of step size parameter, μ, the following formulae will be obtained for the step size parameter [1]:

$$\mu(i) = \mu_{max} \cdot [\,\|\mathbf{p}_i\|^2 / \|\mathbf{p}_i\|^2 - C] \qquad (4)$$

where C is a small positive constant, α is positive constant close to 1 and $\|\ .\ \|$ denote the norm operator.

The parameter pi can be estimated by time averaging as in equation (5) [1]:

$$\mathbf{p}_i = \alpha\, \mathbf{p}_i + (1-\alpha)\, \mathbf{U}^*_i (\mathbf{U}^*_i \mathbf{U}_i)^{-1} \mathbf{e}_i \qquad (5)$$

[1] POLITEHNICA University of Bucharest, Electronic and Telecommunications Faculty
Telecommunication Department, sorin@elcom.pub.ro

III. THE IMPLEMENTATION AND MAIN RESULTS

The new algorithm was implemented on an evaluation kit with DSP processor (ADSP21161 EZ-KIT).

The ADSP-21161 SHARC DSP is the first low-cost derivative of the ADSP-21160 featuring Analog Devices' Super Harvard Architecture. Like other SHARCs, the ADSP-21161 is a 32-bit processor that is optimized for high performance DSP applications. The ADSP-21161N includes a 100 MHz core, a dual-ported on-chip SRAM, an integrated I/O processor with multiprocessing support, and multiple internal busses to eliminate I/O bottlenecks.

The ADSP-21161 offers a Single-Instruction-Multiple-Data (SIMD) architecture. Using two computational units the ADSP-21161 can double cycle performance on a range of DSP algorithms.

Figure 1 shows a block diagram of the ADSP-21161, illustrating the following architectural features:

- Two processing elements, each made up of an ALU, Multiplier, Shifter and Data Register File
- Data Address Generators (DAG1, DAG2)
- Program sequencer with instruction cache
- PM and DM buses capable of supporting four 32-bit data transfers between memory and the core every core processor cycle
- Interval timer
- On-Chip SRAM (1 Mbit)
- SDRAM Controller for glueless interface to SDRAMs
- External port that supports:
- Interfacing to off-chip memory peripherals
- Glueless multiprocessing support for six ADSP-21161N SHARCs
- Host port read/write of IOP registers
- DMA controller
- Four serial ports
- Two link ports
- SPI-compatible interface
- JTAG test access port
- 12 General Purpose I/O Pins

The main characteristics of the ADSP21161 are presented below:

- High performance 32-bit DSP—applications in audio, medical, military, wireless communications, graphics, imaging, motor-control, and telephony
- Super Harvard Architecture-four independent buses for dual data fetch, instruction fetch, and nonintrusive, zero-overhead I/O

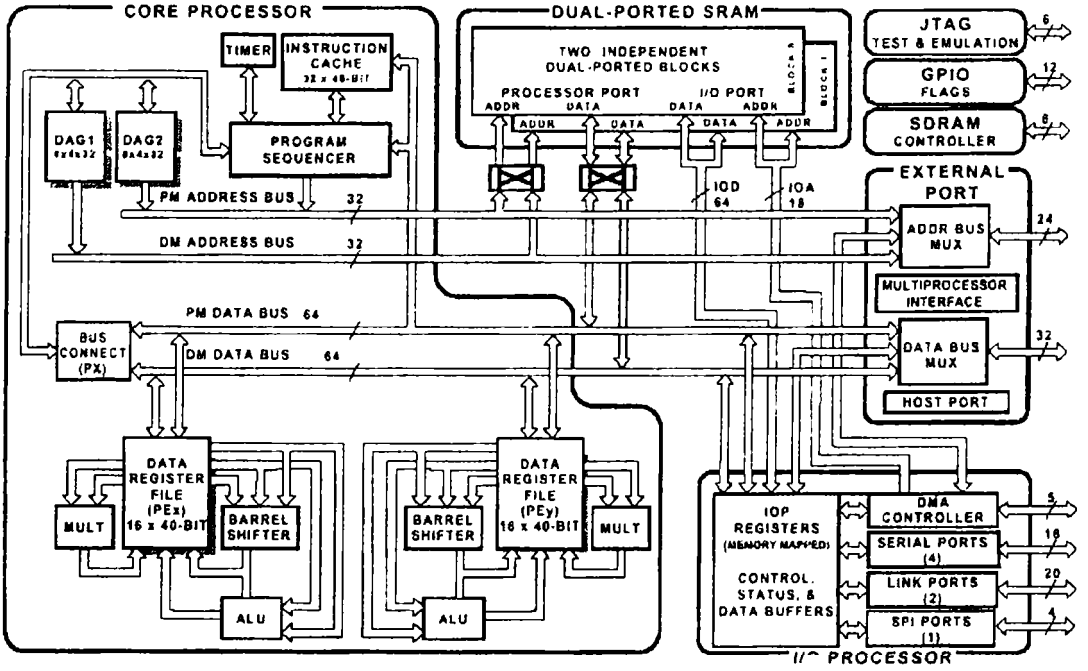


Figure 1. The block diagram of the ADSP-21161

- Single-Instruction-Multiple-Data (SIMD) computational architecture- two 32-bit IEEE floating-point computation units, each with a multiplier, ALU, shifter, and register file
- Serial ports offer I 2 S support via 8 programmable and simultaneous receive

and transmit pins, which supports up to 16 transmit or 16 receive channels of audio

- Integrated peripherals—integrated I/O processor, 1 Mbit on-chip dual-ported SRAM,SDRAM controller, glueless multiprocessing features, and I/O ports (serial, link, external bus, SPI, & JTAG)
- ADSP-21161N supports 32-bit fixed, 32-bit float, and 40-bit floating point formats.
- 100 MHz (10 ns) core instruction rate
- Single-cycle instruction execution, including SIMD operations in both computational units
- 600 MFLOPS peak and 400 MFLOPs sustained performance
- 1 Mbit on-chip dual-ported SRAM (0.5 Mbit block 0, 0.5 Mbit block 1) for independent access by core processor and DMA
- 400 million fixed-point multiply and accumulation operations (MACs) sustained performance
- Dual Data Address Generators (DAGs) with modulo and bit-reverse addressing
- Zero-overhead looping with single-cycle loop setup, providing efficient program sequencing
- IEEE 1149.1 JTAG standard test access port and on-chip emulation
- Single Instruction Multiple Data (SIMD) architecture provides:
    1. Two computational processing elements
    2. Concurrent execution--Each processing element executes the same instruction, but operates on different data
- Parallelism in busses and computational units allows:
    1. Single-cycle execution (with or without SIMD) of: a multiply operation, an ALU operation, a dual memory read or write, and an instruction fetch
    2. Transfers between memory and core at up to four 32-bit floating- or fixed-point words per cycle, sustained 1.6 Gigabytes/second bandwidth
    3. Accelerated FFT butterfly computation through a multiply with add and subtract
- DMA Controller supports:
    1. 14 zero-overhead DMA channels for transfers between ADSP-21161N internal memory and external memory, external peripherals, host processor, serial ports, link ports or Serial Peripheral Interface (SPI) interface
    2. 64-bit background DMA transfers at core clock speed, in parallel with full-speed processor execution
    3. 800 Mbytes/s transfer rate over IOP bus
    4. Host processor interface to 8-, 16- and 32-bit microprocessors, the host can directly read/write ADSP-21161 IOP registers.
- 32-bit (or up to 48-bit) wide synchronous External Port provides glueless connection to asynchronous, SBSRAM and SDRAM external memories
- Memory interface supports programmable wait state generation and wait mode for off-chip memory
- 24-bit address, 32-bit data bus. 16 additional data lines via multiplexed link port data pins allow complete 48-bit wide data bus for single-cycle external instruction execution
- 32-48, 16-48, 8-48 execution packing for executing instruction directly from 32-bit, 16-bit, or 8-bit wide external memories
- 32-48, 16-48, 8-48, 32-32/64, 16-32/64, 8-32/64, data packing for DMA transfers directly from 32-bit, 16-bit, or 8-bit wide external memories to and from internal 32-, 48-,or 64-bit internal memory
- SDRAM Controller for glueless interface to low cost external memory
- Extended external memory banks (64 M-words) for SDRAM accesses
- Multiprocessing support provides: glueless connection for scalable DSP multiprocessing architecture and distributed on-chip bus arbitration for parallel bus connect of up to six ADSP-21161s. global memory and a host
- Two 8-bit wide link ports for point-to-point connectivity and array multiprocessing be-tween ADSP-21161
- Serial Ports provide:four 50 Mbit/s synchronous serial ports with companding hardware, 8 bi-directional serial data pins, configurable as either a transmitter or receiver, TDM support for T1 and E1 interfaces, and 128 TDM channel support

The VSS-AP algorithm was implemented and compared with the LMS algorithm under the following circumstances: the input signal was chosen as a colored signal (that is, filtering a white, zero-mean, Gaussian random sequence through a second order IIR filter), K=4, M=17, α = 0,995, C = 10^{-4}, μ_{max} = 1.0 .

The main results, illustrated in figure 2, show a significantly performance improved for the VSS-AP (both for convergence speed and misadjustment). The execution time is quite reasonable. Using an adequate technique, such as switching buffers, a real time implementation is realizable.

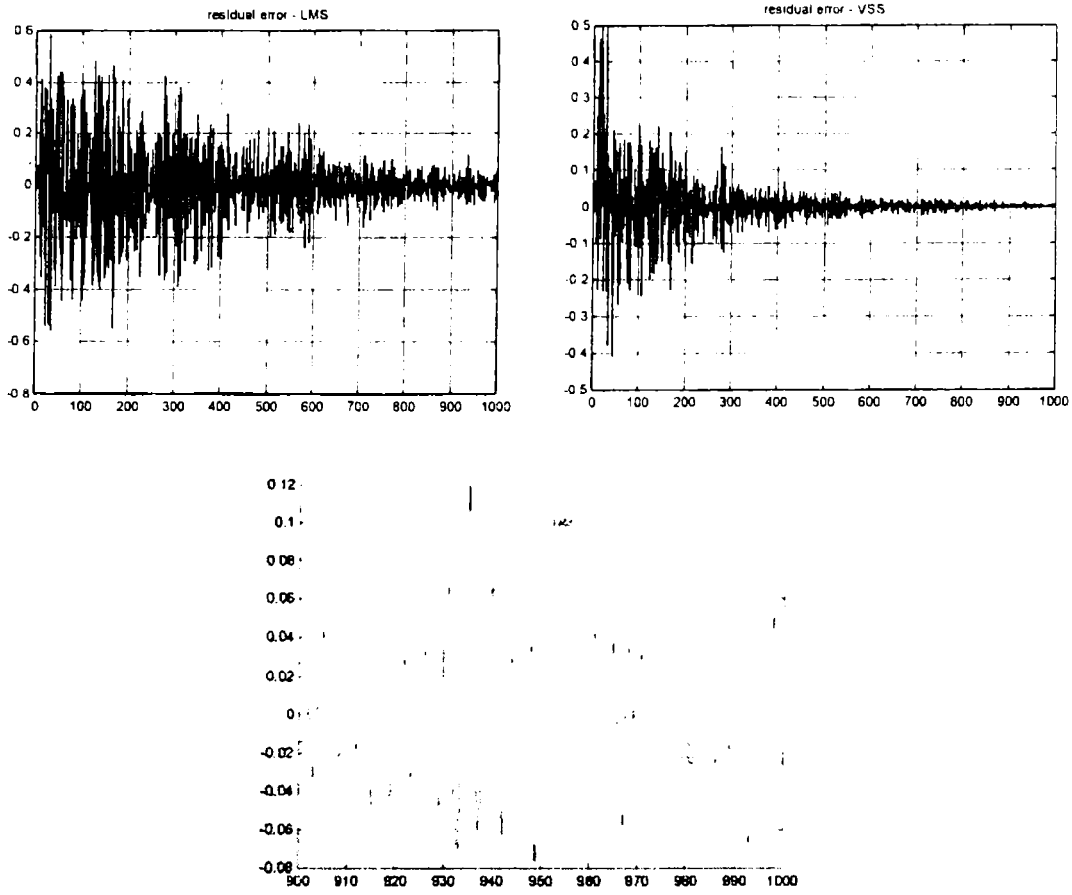
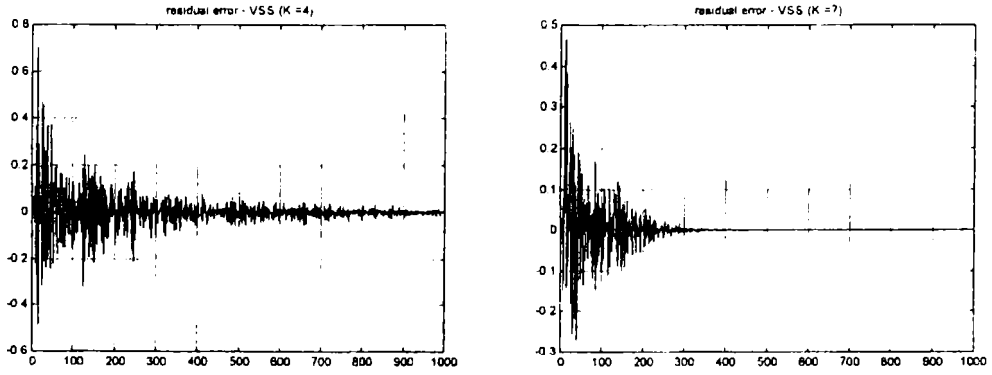


Figure 4. The main results (K=4)



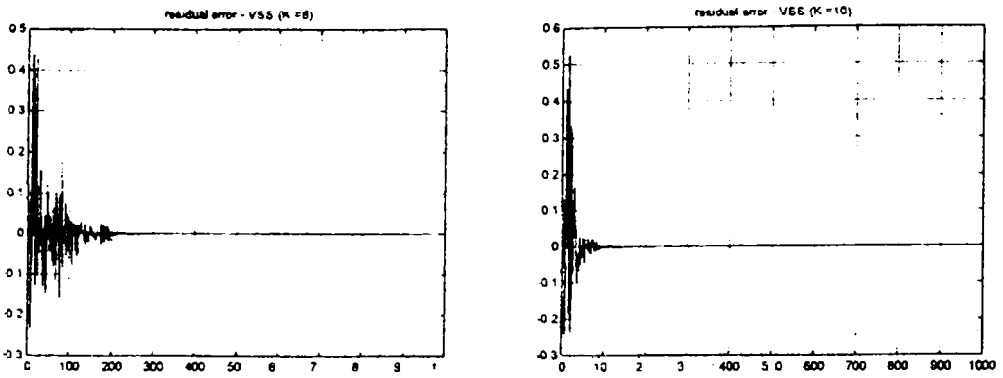


Figure 3. The VSS adaptive filter performance for various values of parameter K

If the dimension of the input matrix is increased then the performance of the adaptive filter is better. For K large enough the performance is quite similar to RLS algorithm. In order to evaluate the inverse of the matrix U we use a Gauss-Jordan inversion algorithm witch was implemented in C language.

The execution time (in processor cycles) is illustrated for several values of the parameter K in figure 4.

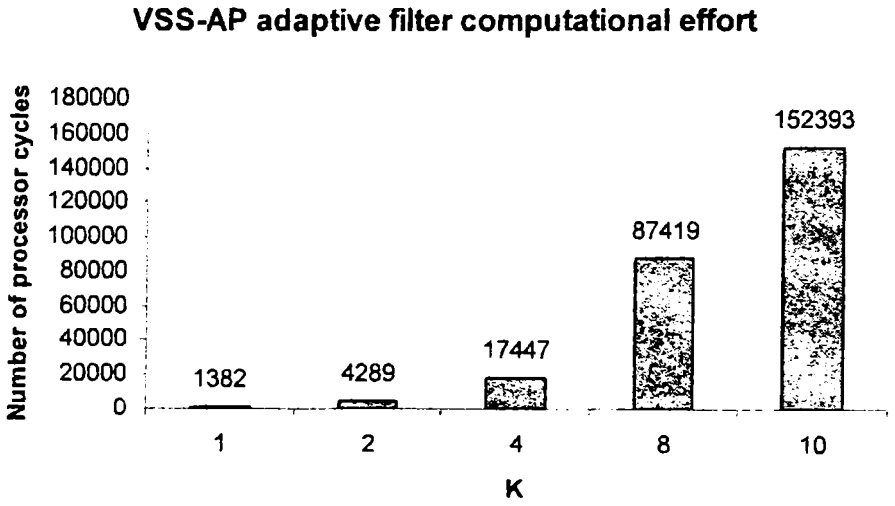


Figure 4. The computational effort for VSS-AP implementation

The VSS-AP implementation takes an execution time between and 0.2 ms and 16 ms witch is a quite reasonable execution time if the switching buffer technique is involved. This techniques requires as a timing condition that the acquisition time for the whole signal window (about 20 ms) must be greater than total execution time for signal processing in the current window.

IV. CONCLUSIONS

A new adaptive algorithm was presented. The performance of the new algorithm is better than classical algorithm (similar with RLS algorithm) but the computational effort is smaller than RLS.

This algorithm woks very well for ill-conditioned input signal (e.g. speech signal).

V. REFERENCES

[1] Shin, H.C, et. al, "Variable Step Size NLMS and Affine Projection Algorithms" , IEEE Signal Processing Letters, vol.11, no. 2, february 2004, pp. 132-135

[2] S. Haykin, "Adaptive Filter Theory", 3rd ed., Prentice Hall, 1996

[3] ADSP2116x Hardware Reference Manual, Analog Devices 2000

[3] ADSP2116x Software Manual, Analog Devices 2000