U.P.B. Sci. Bull., Series C, Vol. 71, Iss. 4, 2009 ISSN 1454-234x

ECHO CANCELLATION USING THE LMS ALGORITHM

Cristina Gabriela SĂRĂCIN[1], Marin SĂRĂCIN[2], Mihai DASCĂLU[3], Ana-Maria LEPAR[4]

O problema majoră în comunicațiile din Internet (messenger de voce, VoIP) este ecoul produs de microfoanele utilizate şi latența mare a transmisiei. Scopul lucrării este de a prezenta algoritmul LMS (un algoritm adaptiv). Simulările anulării zgomotului / ecoului sunt realizate in Matlab. Din punctul de vedere al implementării hardware a algoritmului, a fost utilizat un procesor DSP (Digital Signal Processor) din cadrul chitului de dezvoltare de la SHARC (ADSP-21061).

A major problem in voice communication over the Internet (voice messenger, VoIP) is represented by the created echo, due to the microphones used and the very high latency of the transmission. The goal of this paperwork is to present the LMS algorithm (an adaptive algorithm). The simulations of the cancellation of noise / echo are done in Matlab software. Regarding the hardware implementation of the algorithm, a DSP processor (Digital Signal Processor) from SHARC development kit (ADSP-21061) was used.

Keywords: numerical filters, adaptive filters, LMS, adaptive cancellation of echo

1. Introduction

The paper presents a solution to noise / echo cancellation and a hardware real-time implementation of the LMS algorithm.

Although other researches have been made in this field ([2], [3]), the results obtained from this implementation are optimistic and with future optimizations regarding the used algorithm, the output can be improved. Also hardware and real-time processing constraints had to be taken into consideration.

The following sub-sections will present a brief description of numerical and adaptive filters focused on the cancellation of noise and echo. The second section if focused on the LMS algorithm and its simulation in Matlab. The third section presents the simulation results obtained from the adaptive cancellation of noise. A brief description of the DSP and hardware implementation of the algorithm is discussed in the fourth section.

[1] Lecturer. Dept.of Electical Engineering, University POLITEHNICA of Bucharest, Romania
[2] Prof., Dept.of Electical Engineering, University POLITEHNICA of Bucharest, Romania
[3] Student, Dept.of Computer Science, University POLITEHNICA of Bucharest, IT Consultant, CCT S.R.L, Bucharest, Romania
[4] Student, Dept.of Computer Science, University POLITEHNICA of Bucharest, Romania

2. Filters – general presentation

2.1. Numerical Filters

The analog filter represents a functional bloc with selective proprieties in the area of frequencies. If the signal x(t) has a given area of frequencies, then, because of the filter, in the area of the signal y(t) only a part of the original signal x(t) with the same amplitudes will be found, and the rest will not be taken into consideration. In this case, the functional bloc has selective proprieties of some frequencies from the area of the entrance signal.

Starting from the continuous signal, a system with the same functionality as the analog filter, but within the area of discrete signals is used ([1]).

In this case, the functional block represents a computing algorithm, which generates a sequence of data as outputs starting from a given sequence of data as input.

Most filters are implemented and simulated using Matlab.

Generally, the functionality of Matlab for generating random signals is created using the following functions ([4]):

```
u = idinput(N)
u = idinput(N,type,band,levels)
[u,freqs] = idinput(N,'sine',band,levels,sinedata),
```

where *type* can be ‘rgs’ – random number, Gaussian signal, ‘rb’: random, binary signal , ‘prbs’ –pseudo random binary signal , ‘sine’: signal as a sum of sinusoids.

2.2. Adaptive Filters. WIENER filtration

The Wiener filter consists of a digital input signal x(k), an desired output d(k) and a linear filter containing the answer to the impulse h(k); the input x(k) is used to obtain the output y(k) ([6]). The difference between the output of Wiener filter y(k) and the desired output d(k) is the estimation error e(k).

The importance of Weiner filtration is given by the answer of the impulse h(k), which gives a small estimation error. So, the main purpose is to minimize the square value of e(k).

The Wiener filters are FIR filters (Finite Impulse Response). One of the proprieties of FIR filters is the linear phase, which means that the signal of the passant band will not suffer any dispersion. This phenomenon appears when different frequency components of the signal have different delays in the system.

An FIR filter is a system permanently fed. Its transfer function is:

$$H(z) = \sum_{k=-\infty}^{\infty} h[n] \cdot z^{-k} \tag{1}$$

So the output can be expressed using the following equation:

$$H(z) = b_0 + b_1 \cdot z^{-1} + b_2 \cdot z^{-2} + \ldots + b_n \cdot z^{-n} \tag{2}$$

The Wiener filter is a N length causal filter and it is the most famous adaptive structure. Its configuration is presented in the following diagram:

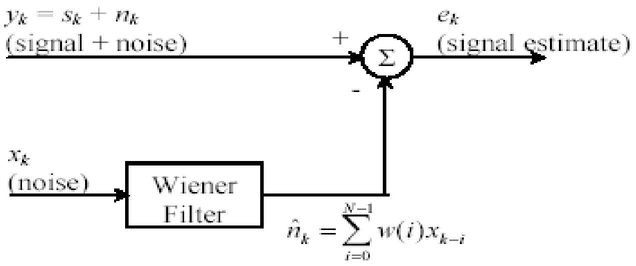

Fig. 1. The configuration of Wiener Filter

The Nth sample of the Y signal, called Y_k consist of two components: the main signal S_k and a noise N_k, which is related to X_k.

The Wiener filter offers an optional estimation of N_k, $\hat{n}_k$. The estimated error e_k is obtain from the extraction of the noise estimation Wiener $\hat{n}_k$ from the input signal y_k.

$$e_k = y_k - \hat{n}_k = y_k - \sum_{i=0}^{N-1} w(i) \cdot x_{k-i} \quad (3),$$

where $y(k) = \sum_{n=0}^{N-1} h_n x(k-n)$ and $w(i)$ are the coefficients of the Wiener filter.

2.3. Adaptive Filtering

The main role of adaptive filtering is the development of a filter capable of adjusting to the statistics of the signal. Usually, an adaptive filter takes the form of a FIR filter, with an adaptive algorithm that modifies the values of its coefficients.

There are three factors that measure the efficiency of an adaptive algorithm:

- The complexity of calculus measurements and the amount of calculus executed at each step;
- The speed of adjustment that allows an adaptive filter to converge to the Weiner solution;
- The estimation error obtained from the difference between the present Weiner solution and the solution given by the adaptive algorithm.

The main configurations of adaptive filters are the adaptive cancellation of noise and the adaptive cancellation of echo.

2.4. The adaptive cancellation of noise

In this configuration, the input signal x(n) and a noise source $N_1(n)$ are compared with a desired signal d(n), which consists of a signal s(n) distorted by another noise $N_0(n)$.

The coefficients of the adaptive filter are adjusted in order to reduce the error e(n) to the optimal value: 0.

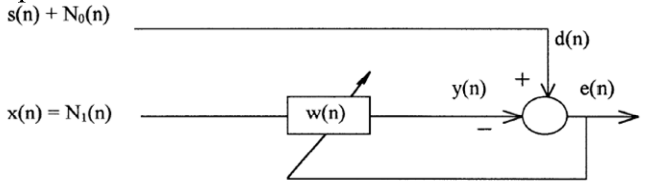

Fig. 2. The block schema for the adaptive cancellation of noise

Both noise signals for this configuration must not be related with s(n) signal. Furthermore, the two noise signals should be related one to each other, which means they can come from the same source. The error signal will converge to zero.

2.5. The adaptive cancellation of echo

The configuration generates a replica of the echo at the output y(n) which decreases from the acoustic echo d(n). The error e(n) is obtained from the difference between them.

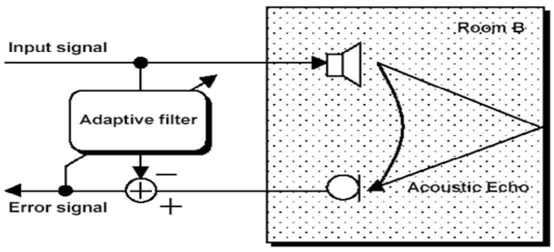

Fig. 3. The bloc schema for the adaptive cancellation of the echo

3. The LMS algorithm

The LMS algorithm is one of the most used algorithms because it is easy and stable ([9]). The only disadvantage is its weak convergence. Two inputs are required:

- A reference noise that should be related with the noise that exists in distorted the input signal. This means that the noise comes from the same source.

- An error signal already calculated.

In Matlab / Simulink there already exists a dedicated block ([5], [6]):

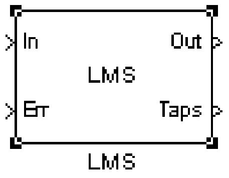

Fig. 4. The schema of the adaptive filter in Matlab

The LMS adaptive filter implements an FIR adaptive filter using the NLMS algorithm (Normalized Least Mean-Square). A short description is provided below:

$$y(n) = \hat{w}^H(n-1) \cdot u(n)$$
$$e(n) = d(n) - y(n) \qquad (4),$$
$$\hat{w}(n) = \hat{w}(n-1) + \frac{u(n)}{a + u^H(n) \cdot u(n)} \mu e^*(n)$$

where: n – the present step of the algorithm; u(n) – the input at step n; w(n) – the array with the values of adaptive filter at step n; y(n) – the filtered output at step n; e(n) – the estimated error at step n; d(n) – the desired answer at step n; μ – the step to adjust; a – positive constant for reducing the numeric instability.

The obtained results were:

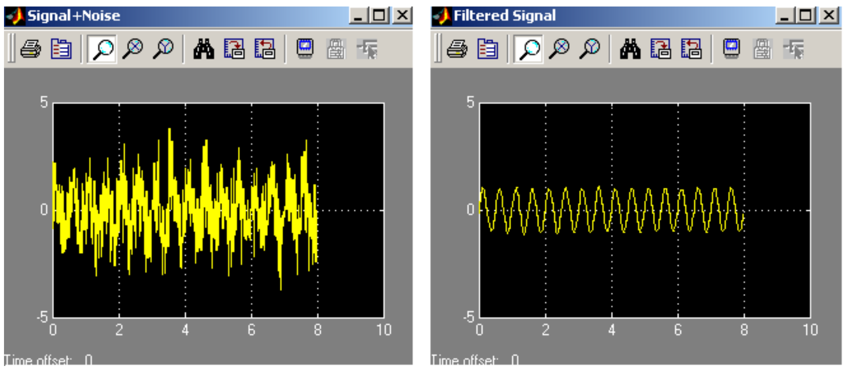

Fig. 5. The distorted input and the result

The length of the FIR filter is 10 and the size of the step is 0.001. The configuration works if an uniform noise is used.

4. The adaptive cancellation of the noise

As an input signal a wav file was used together with a Gaussian noise, with a period of 0.001 seconds (Fig. 6). The results are presented in Fig. 7:

The length of FIR filter is 8 and the size of the step is 0.01.

In the simulation process two microphones were used, one for the noise reference signal and another for the distorted one (useful signal + noise). The microphones are connected to the entry of a stereo sound card (Line In) of a computer. The nLMS block from Matlab generates a 128x2 matrix and in order to divide the two channels a sub-matrix block is used. The signal is captured at a sampling rate of 8kHz, stereo, 128 frames per frames and has a width of 16 bits.

Because in the Matlab implementation were processed each time, the output signal has a delay for each stream of data. If the number of read samples is reduced, the signal is no longer clear as the delay is more pronounced.

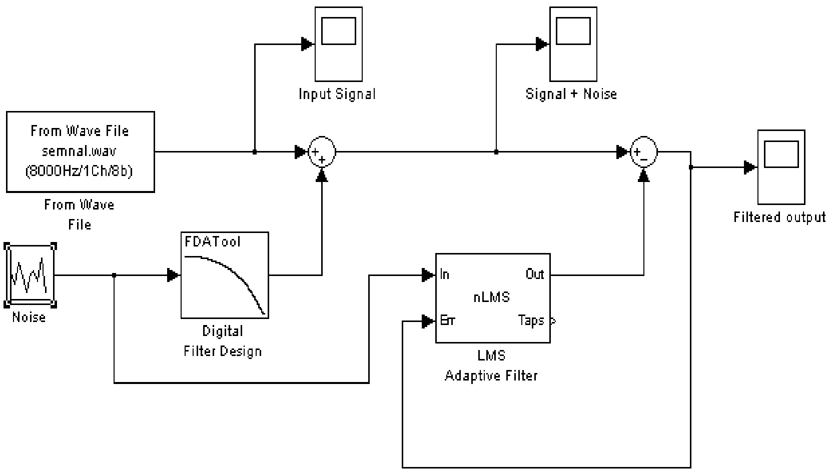

Fig. 6. The block diagram of the adaptive noise cancellation generated by a wave signal

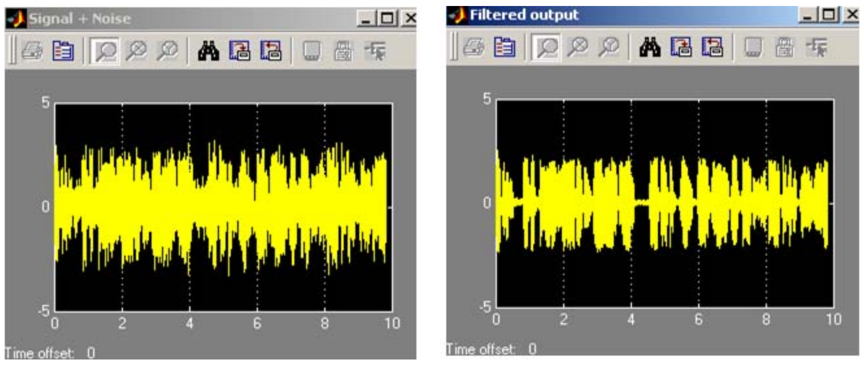

Fig. 7. The distorted signal and the result

5. DSP

DSP comes from Digital Signal Processing and represents a family of applications that test a large area of signals (video, audio etc) using a digital microprocessor also known as *Digital Signal Processor*.

In order to obtain compulsory performances, the signal processors must have parallel multiplications and additions, multiple accesses to their memory and the possibility to efficiently generate the addresses. DSPs also contain a lot of registries for temporary retaining of data and special properties as delays and circular addressing. The DSP consists of three mathematic operations: additions, multiplications and delays.

In Harvard architecture, the DSP has two separate buses of memory which allow simultaneous access to two memory locations. One of the buses is used for reading the code of the instruction, and the other is used for reading the operands. Because three pieces of information are needed, the Harvard–DSP includes a cache memory. The main purpose of this memory is to store the common instructions most likely to be reused, leaving both buses free for the operands. This extension is called *Super Harvard Architecture* (SHARC).

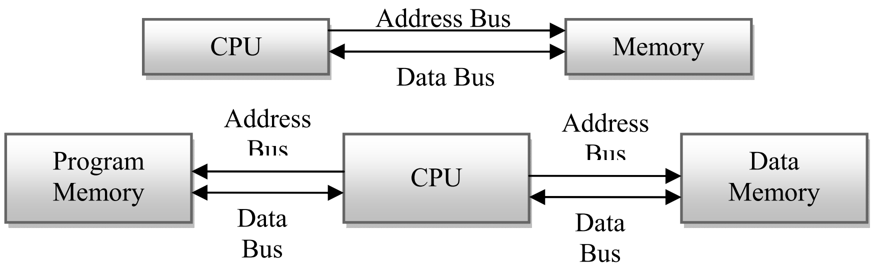

Fig. 8. Super Harvard Architecture

The systems containing DSPs should function in real time, capturing and processing information during the changes of the input signal. First of all, the analog-to-digital converter should keep up with de continuous modifications of real data. Remaining behind means that information is lost and the signal is distorted. For implementation, a development kit from SHARC based on **ADSP-21061** was used**.**

The program was written in Visual DSP++ and represents an implementation of an adaptive algorithm for cancellation of echo. The development environment is able to write assembly code directly in the DSP memory via the serial port connection. The synchronous communications on the serial port are easy to implement because of the simplicity of the used protocol and the maximum data rate is of 40Mbps because the serial ports can operate at the full clock rate of the processor.

The adaptive filter was based on the LMS algorithm. The echo was generated by drawing 50% from the current sample and 25% from two previous samples. The coefficients of the adaptive filter were updated in correlation with the transmitted and received data. The input of the program is a voice with echo and the output converges to a signal without echo.

6. Conclusions

The adaptive cancellation of the noise has many applications, because interferences are common for many environments. Besides the LMS algorithm, there are other algorithms like RLS and Kalman, more powerful and with a higher convergence speed. These algorithms ([11] and [12]) or other approaches ([10]) will be taken into consideration in further developments. Another configuration is the linear adaptive intensification, but the required order of filters is too high and better hardware is needed in terms of computational power. Other promising paths of research would be to use this algorithm combined with the compression of useful information similar with [8] or with wavelet function representations in a new methodology from [13].

All configurations need a reference noise which is related to the noise that distorted the original signal. In order to generate the noise, only one microphone was used. This proved to be difficult, because the noise's wide frequency spectrum.

The overall performance of the designed adaptive filter compared with other implemented systems using the same filter is good, and, with further improvements the results will improve. The LMS algorithm provides good numerical stability and its hardware requirements are low, therefore being the best choice on the available hardware platform. A disadvantage of this algorithm is its weak convergence, but based on obtained results this didn't prove to become a major issue. On the other hand, the NLMS algorithm is one of the most implemented adaptive algorithms in actual telecom/industrial applications.

The echo cancellation algorithm is recommended to be implemented in hardware in case of Internet conversations (VoIP or voice chat) where high latencies and echo can induce perturbations.

R E F E R E N C E S

[1]. *Glen Zelniker*, *Fred J. Taylor*, Advanced Digital Signal Processing (Theory and Applications), DekKer Book Company, 1994

[2]. *Andreas Antoniou*, Digital filters analysis and design, McGraw-Hill Book Company, 1979

[3]. *Marc Moonen*, "Introduction to Adaptive Signal Processing", ESAT/SISTA, Leuven, Belgium, Part III: FIR Adaptive Filter Algorithms, Chapter 7 Fast RLS algorithms, pp 143-189

[4]. http://www.staff.ncl.ac.uk/oliver.hinton/eee305/index.htm

[5]. *Sophocles J. Orfanidis*, Introduction to Signal Processing, Pearson Education, 1995

[6]. http://hermes.etc.upt.ro/teaching/ps/

[7]. http://cnx.org/content/m10481/latest/#fi

[8]. *Dan Stefanoiu, Janetta Culita*, Signal Compaction By Maximum Verisimilitude, U. P. B. Sci. Bull., Series C, **Vol. 70**, No. 3, 2008

[9]. *Bryan Davis*, Adaptive Noise Cancellation using LMS and optimal filtering, University of Florida, available online at http://plaza.ufl.edu/badavis/EEL6502_Project_1.html

[10]. *V. R. Vijaykumar, P. T. Vanathi, P. Kanagasapabathy*, Modified Adaptive Filtering Algorithm for Noise Cancellation in SpeechSignals, Electronics and Electrical Engineering, ISSN 1392 – 1215, no 2, 2007

[11]. *Ying He, Hong He, Li Li, Yi Wu, Hongyan Pan*, The Applications and Simulation of Adaptive Filter in Noise Canceling, 2008 International Conference on Computer Science and Software Engineering

[12]. *J. Ondracka, R. Oravec, J. Kadlec, E. Cocherová*, Simulation of RLS and LMS algorithms for adaptive noise cancellation in Matlab, available at http://dsp.vscht.cz/konference_matlab/matlab00/ondracka.pdf

[13]. *D. Olaru, M. O. Popescu* Singular perturbation detection using wavelet function representation, U. P. B. Sci. Bull., Series C, **Vol. 70**, No. 2, 2008.