Sborník vědeckých prací Vysoké školy báňské - Technické univerzity Ostrava
číslo 2, rok 2005, ročník LI, řada strojní
článek č. 1474

Marek KUPCZAK*

USING THE DISCRETE WAVELET TRANSFORMS FOR DE-NOISING IN DSP

POUŽITÍ DISKRÉTNÍCH WAVELETOVÝCH TRANSFORMACÍ PRO ODSTRANĚNÍ ŠUMOVÉ SLOŽKY ZE SIGNÁLU V DSP

Abstract

This contribution describes the results of an implementation of discrete wavelet transforms (DWTs) into a digital signal processor (DSP). In particular for the realisation, there was used a 32-bit DSP with floating-point format – ADSP-21065L SHARC DSP.

Designed algorithms, based on DWTs, were realized in C language and in assembly language both. These algorithms were used for de-noising of non-stationary signals. The hard threshold is calculated as Johnstone-Donoho universal threshold of the estimated value of noise standard deviation on the first decomposition level.

Abstrakt

Tento příspěvek popisuje výsledky implementace diskrétních waveletových transformací (DWTs) do digitálního signálového procesoru (DSP). Konkrétně pro realizaci byl použit 32bitový ADSP-21065L SHARC DSP procesor s možností práce dat ve formátu plovoucí řádové čárky.

Navržené algoritmy, založené na DWTs, byly realizovány pomocí jazyka C a asembleru. Byly použity pro aplikaci, která je určená pro odstraňování šumu z nestacionárních signálů. Práh pro tvrdé prahování byl vypočten jako Johnstone-Donoho univerzální práh z odhadu velikosti směrodatné odchylky bílého šumu na první úrovni rozkladu.

1 INTRODUCTION

In these days for the DSP applications developing, C language and assembly language are used together. VisualDSP++ is the software product and it contains an integrated environment for developing and debugging of applications. The user interface provides the access to tools for a code developing. Among these tools belong C/C++ compiler, assembler, linker, libraries containing the mathematical functions, functions for digital signal processing, I/O functions and so on.

The C/C++ library is the collection of functions, macros and class templates that can be called from the source files. Concretely, C/C++ library for the ADSP-21xxx was used. The library can divided into four partial libraries: C library, C++ library, DSP library, I/O library. These libraries contain functions for a memory allocation, functions for the char and string routines, mathematical functions, functions for the digital signal processing, I/O functions.

DSP library contains functions for the digital signal processing. It contains functions for the computation of coherence, covariance and correlation, function for the computation with the complex data, functions for the filters, functions for the computation of the direct and the inverse Fast Fourier Transform and functions for the computation with the time windows.

* Ing, Department of Control Systems and Instrumentation, Faculty of Mechanical Engineering, VŠB - Technical University of Ostrava, 17. listopadu, Ostrava-Poruba, tel. (+420) 59 732 1280, e-mail marek.kupczak.fs@vsb.cz

But this library doesn't contain functions for the signal processing and the signal analysis based on the discrete wavelet transforms (DWTs).

2 DESIGNED AND REALIZED ALGORITHMS BASED ON DWTS INTO DSP

The discrete wavelet transforms are very popular and many contributions on this subject were published. So in this work the detail description of these transformations isn't presented. There are presented only own results of the implementation into SHARC DSP. Discrete wavelet transforms are linear, so they can be defined by matrix form. When a transform matrix $\mathbf{W}$ is an orthogonal, an inverse matrix is transposed $\mathbf{W}^{-1} = \mathbf{W}^{\mathbf{T}}$ (only for real matrices).

For post-processing of signals we can choose a record length and further we can create a sparse matrices for direct and inverse DWTs. Digital signal processor are dedicated for real-time processing, so we cannot take for example 1024 samples and do an analysis on them. We have to define these sparse matrices in the infinite form, given by form (1) and form (2). The filters length is in this case six. The low pass filter is $\mathbf{h}$ and the high pass filter is $\mathbf{g}$.

$$\mathbf{W} = \begin{pmatrix} h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & 0 & 0 & \dots & 0 \\ g_0 & g_1 & g_2 & g_3 & g_4 & g_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & g_3 & g_4 & g_5 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 & g_4 & g_5 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & g_0 & g_1 & g_2 & g_3 & g_4 & g_5 & 0 & 0 & \dots & 0 \end{pmatrix} \tag{1}$$

$$\mathbf{W}^{-1} = \begin{pmatrix} h_0 & 0 & 0 & 0 & g_0 & 0 & 0 & 0 \\ h_1 & 0 & 0 & 0 & g_1 & 0 & 0 & 0 \\ h_2 & h_0 & 0 & 0 & g_2 & g_0 & 0 & 0 \\ h_3 & h_1 & 0 & 0 & g_3 & g_1 & 0 & 0 \\ h_4 & h_2 & h_0 & 0 & g_4 & g_2 & g_0 & 0 \\ h_5 & h_3 & h_1 & 0 & g_5 & g_3 & g_1 & 0 \\ 0 & h_4 & h_2 & h_0 & 0 & g_4 & g_2 & g_0 \\ 0 & h_5 & h_3 & h_1 & 0 & g_5 & g_3 & g_1 \\ 0 & 0 & h_4 & h_2 & 0 & 0 & g_4 & g_2 \\ 0 & 0 & h_5 & h_3 & 0 & 0 & g_5 & g_3 \\ 0 & 0 & 0 & h_4 & 0 & 0 & 0 & g_4 \\ 0 & 0 & 0 & h_5 & 0 & 0 & 0 & g_5 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{2}$$

The newly designed algorithms, based on DWTs, reflect the fact that these matrices are sparse. The first decomposition step of an analyzed signal is defined as $\mathbf{Y} = \begin{pmatrix} \mathbf{cA_1} \\ \mathbf{cD_1} \end{pmatrix} = \mathbf{W} \cdot \mathbf{y}$, where $\mathbf{cA}_1$ is a vector of approximation coefficients and $\mathbf{cD}_1$ is a vector of detail coefficients. But in fact we could execute relatively few operations of multiply-accumulate cycles because for example the individual coefficients of the decomposition we can calculate by following form (3)

$$cA_i = \sum_{j=0}^{K-1} h_j \cdot y_{pozice+j}$$
$$cD_i = \sum_{j=0}^{K-1} g_j \cdot y_{pozice+j} \tag{3}$$

To calculate each individual coefficient of signal decomposition or reconstruction, we don't have to multiply the huge ***sparse*** matrix with an input data vector or matrix, but it's possible to calculate these coefficients as the multiply-accumulate cycle between the filter coefficients and the individual data vector or matrix, respectively. The length of the total sum of the operations for the computation of each approximation or detail coefficient isn't *N* (ordinarily hundreds or thousands of data), but is same with the length of the filter vectors **h** and **g** (ordinarily units of data). By this approach we obviate redundant operations like the multiplication with zero elements of the transform matrix **W** or $\mathbf{W}^{-1}$, respectively.

There were realized two functions which are a core for the next example, concretely for de-noising of non-stationary signals. These functions perform one step of signal decomposition and reconstruction. Their names are *dwt* (discrete wavelet transform) and *idwt* (inverse discrete wavelet transform). The definitions of these functions are in C language, but the declaration of them is in assembly language. The function declarations for direct and inverse discrete wavelet transform are following:

float dwt (*pm float *coefficients*, *dm float delay_line*, *int filter_length*, *int index*)

float idwt (*pm float *coefficients*, *dm float delay_line*, *int filter_length*, *int index*)

These functions have four input arguments and one output argument (a computed result). The first input argument is a pointer on a filter coefficients array, the second input argument is a pointer on a delay line array of measured of computed samples, the third input argument is low pass or high pass filter length and the fourth is an index that indicates a place for which a computation has to begin.

Filter coefficients are stored in DSP' program memory (*pm*), measured or computed samples are stored in DSP' data memory (*dm*). The fourth input argument is necessary because a computation isn't performing in each step; in fact this is a multirate system with different decimation factors.

3 MULTIRESOLUTION DECOMPOSITION AND THRESHOLDING

The decomposition process can be iterated, with successive approximations being decomposed in turn, so that one signal is broken into many lower-resolution components. This process can be shown in Fig. 1.

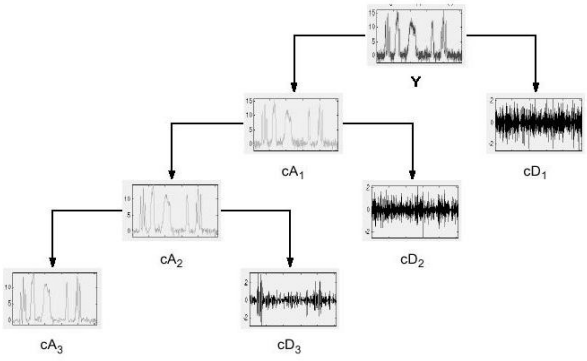


Fig. 1 Multiresolution decomposition

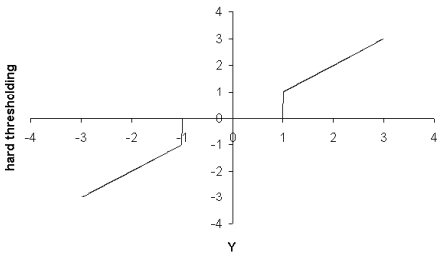
The thresholding is the simplest wavelet non-linear shrinkage technique. It is common for all thresholding rules to set to 0 the coordinates of a vector $\mathbf{Y}$ which is subjected to thresholding, if they are smaller in absolute value than a fixed non-negative number – the threshold λ. Depending on how the coordinates of $\mathbf{Y}$ are processed when they are larger than λ one can define different thresholding policies. The two most common thresholding policies are hard and soft. The analytic expressions for the hard- and soft-thresholding rules are (4) a (5).

$$\delta^h(Y_i,\lambda) = Y_i \mathbf{1}\left(|Y_i| > \lambda\right) \tag{4}$$

$$\delta^s(Y_i,\lambda) = \left(Y_i - \operatorname{sgn}(Y_i)\cdot\lambda\right) \mathbf{1}\left(|Y_i| > \lambda\right) \tag{5}$$

where:

$\lambda \geq 0$, $\mathbf{Y} = (Y_1,\ldots,Y_N) \in \mathbf{R}$. The rules are depicted in Fig. 2 and Fig. 3.



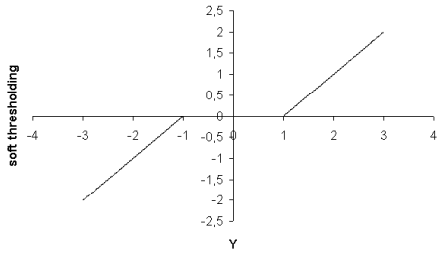


Fig. 2 Hard thresholding rule for $\lambda = 1$ **Fig. 3** Soft thresholding rule for $\lambda = 1$

The optimal thresholding rules are obtained by choosing the threshold $\lambda = \sigma$. Of course, σ is generally not known, and the optimal risk remains unattainable. Assume the following model by form (6), respectively (7)

$$Y_i = \theta_i + \sigma\cdot z_i, \quad i = 1,\ldots,N \tag{6}$$

$$\mathbf{Y} = \boldsymbol{\theta} + \sigma\cdot\mathbf{z} \tag{7}$$

where:

$\mathbf{Y} = (Y_1,\ldots,Y_N) = \mathbf{W}\cdot\mathbf{y}$, $\boldsymbol{\theta} = (\theta_1,\ldots,\theta_N) = \mathbf{W}\cdot\mathbf{s}$ and $\mathbf{z} = (z_1,\ldots,z_N) = \mathbf{W}\cdot\mathbf{e}$. Due to orthogonality of $\mathbf{W}$, $\mathbf{z} \sim \mathrm{N}(0,\sigma^2)$.

The accurate computation of σ is unattainable. But Donoho and Johnstone have shown that the risk of the hard-thresholding rule with a universal threshold $\lambda = \sigma\cdot\sqrt{2\log N}$ is close to the oracular risk.

We can calculate the estimated value of white noise standard deviation. We can propose that the useful signal is mainly low-frequency and the noise is mainly concentrated on the first detail level. The criteria of the standard deviation of the white noise are shown in [Kupczak, 2003]. But we cannot use the whole approach because we have enough time in real-time signal processing operations in DSPs; we have to simplify this approach.

So, we can calculate the estimated value of the standard deviation by this formula (8)

$$\sigma' = \frac{median\left\{\left|cD_1^{(sorted)}\right|\right\}}{0,6745} \tag{8}$$

4 CASE STUDY

Suppose we have to smooth a signal given in Fig. 4. It means to cancel the measurement noise. This signal contains two harmonic signals and the white noise. The standard deviation of the white noise is $\sigma = 2,28$. The sampling frequency is 48 kHz.

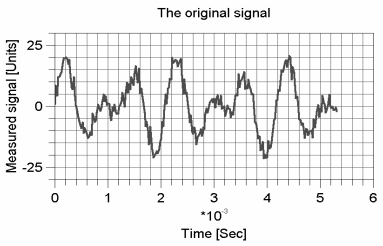


Fig. 4 The original signal

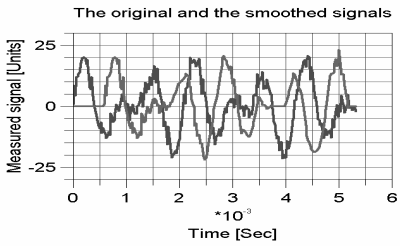


Fig. 5 The original signal and the smoothed signal

In Fig. 2 is shown the original signal and the smoothed signal. In this case, there was used Daubechies basis 3. It means that the low pass and high pass filter length is six. The low pass filter coefficients and the high pass filter coefficients are following:

Tab. 1 Daubechies' 3 LP and HP bases

Low pass filter coefficients	High pass filter coefficients
$h_0 = \ 0,33267$	$g_0 = \ 0,03523$
$h_1 = \ 0,80689$	$g_1 = \ 0,08544$
$h_2 = \ 0,45988$	$g_2 = -0,13501$
$h_3 = -0,13501$	$g_3 = -0,45988$
$h_4 = -0,08544$	$g_4 = \ 0,80689$
$h_5 = \ 0,03523$	$g_5 = -0,33267$

The signal was decomposed to the third level. The computation of the estimated standard deviation was realized for each 32 samples. So it means that the signal can be non-stationary. The values of the estimated standard deviation σ' and threshold λ are following:

Tab. 2 Estimated standard deviation and threshold values

Estimated standard deviation	Thresholding value
$\sigma_0^{'} = 0$	$\lambda_0 = 0$
$\sigma_1^{'} = 1,704$	$\lambda_1 = 4,914$
$\sigma_2^{'} = 2,622$	$\lambda_2 = 7,564$
$\sigma_3^{'} = 1,817$	$\lambda_3 = 5,241$
$\sigma_4^{'} = 1,919$	$\lambda_4 = 5,534$

The sampling frequency of 1819A Codec (Sigma-Delta converter) was set up to 48 kHz. The ADSP-21065L runs at 60 MHz. The delay between the measured and processed sample is 28 steps. So in this case the delay is 0, 6 ms.

5 CONCLUSIONS

This contribution deals with the implementation the discrete wavelet transforms into digital signal processors, in particular into SHARC DSPs. There were designed two algorithms for direct and inverse discrete wavelet transforms for one-dimensional signals. These algorithms were realized in assembly language, but their definitions are in C language. So we can call these functions easily from C interface. These functions are core for one application. This application is dedicated for de-noising of non-stationary one-dimensional signals. In this application the input signal (sampled via 1819A Sigma-Delta Converter) was decomposed via Mallat algorithms into three levels. Further we suppose that the white noise is mostly concentrated on the first decomposition level. From the detail coefficients of this level the estimated standard deviation was calculated. Further from this value, the threshold value was computed as Johnstone-Donoho universal threshold.

This threshold was applicated to the approximation and detail coefficients of the third decomposition level. Further the inverse discrete wavelet transform was used. So, we obtained a smoothed signal. The computation of the standard deviation is applicated in pieces because of non-stationary signal.

The functionality of this application was verified with using an evaluation kit – ADSP-21065L EZ-KIT Lite. This evaluation kit contains 16-bit Sigma-Delta Converter (1819A) and 32-bit ADSP-21065L SHARC DSP with floating-point format that runs at 60 MHz. The sample rate was set up to 48 kHz. In the case study was used Daubechies 3 basis, i. e. the low pass and high pass filter length is six. The delay between a measured sample and a processed sample is (in this case) 28 steps. So it means the delay is less that 1 ms, concretely 0, 6 ms. This small delay (relatively) is given by very short filter length. In conventional application, FIR filters are usually used. In these applications the FIR filter taps are usually very large. So the using of discrete wavelet transforms may be very interesting.

REFERENCES

[1] ČASTOVÁ, N., KALÁB, Z., LYUBUSHIN, A. Discrete wavelet transform analysis of digital records of seismic signals. In *1st International Conference APLIMAT 2002*. Bratislava: Slovak University of Technology, 2002, 6p. ISBN 80-227-1654-5.

[2] DONOHO, D., JOHNSTONE, I. *Ideal de-noising in an orthonormal basis chosen from a library of bases*. Stanford: Department of Statistics, 1994. 8p.

[3] DONOHO, D., JOHNSTONE, I. *Ideal spatial adaptation by wavelet shrinkage*. Stanford: Biometrika, 1994. 425-455p.

[4] KUPCZAK, M. Ideal choice of an orthogonal basis from a library of bases for denoising. In *XXVIII. ASR 2003 Seminar*, VSB – Technical University of Ostrava, 2003, 181p-188p. ISBN 80-248-0326-7.

[5] KUPCZAK, M. The design of the algorithms for the signal processing. In *XXIX. ASR 2004 Seminar*, VSB – Technical University of Ostrava, 2004, 179p-188p. ISBN 80-248-0590-1.

[6] VIDAKOVIC, B. *Statistical modeling by wavelets*. New York: John Wiley & Sons, 1999. 400p. ISBN 0-471-29365-2.

Reviewer: prof. Ing. Jiří Tůma, CSc., VŠB-Technical University of Ostrava