New Algorithm For Calculating Wavelet Transforms

Piotr Lipinski,
Division of Computer Networks, Technical University of Lodz,
Lodz, Stefanowskiego 18/22, 90-924, Poland

and

Mykhaylo Yatsymirskyy
Department of Computer Science, Technical University of Lodz,
Lodz, Wolczanska 215, 90-924, Poland

ABSTRACT

In this article we introduce a new algorithm for computing Discrete Wavelet Transforms (DWT). The algorithm aims at reducing the number of multiplications, required to compute a DWT. The algorithm is general and can be used to compute a variety of wavelet transform (Daubechies and CDF). Here we focus on CDF 9/7 filters, which are used in JPEG2000 compression standard. We show that the algorithm outperforms convolution-based and lifting-based algorithms in terms of number of multiplications.

Keywords : Wavelet transform, Fast wavelet transform

1. INTRODUCTION

Wavelet transforms are widely used in signal compression [1] and coding [2] e.g.: JPEG2000. Unfortunately, the numerical cost per pixel of wavelet compression algorithms is much higher when compared to DCT transforms in JPEG. This is due to the fact that in JPEG the input signal is segmented before compression and the compression of the segments is performed. In wavelet compression algorithms the whole image is transformed, which leads to much higher numerical costs per pixel. Therefore, there is a strong need to develop efficient wavelet transform algorithms.

Wavelet transforms are usually computed using convolution based algorithms (also known as pyramidal algorithm) [3] or ladder algorithms (also known as lift algorithms) [4],[5].

Multiple approaches can be applied to reduce the time and hardware complexity of wavelet algorithms to name only: multiplierless approach [6], direct calculation of wavelet on a fixed level [7], parallel computations [8] dedicated hardware design eg. [9], wavelet filters shifting [10]. In this article, we use yet another approach. We modify the convolution-based algorithm in order to reduce the number of multiplications. This approach can be used to compute the family of Daubechies and CDF wavlets [11]. We take advantage of the factored form of Daubechies and CDF filters so as to transform those filters into the product of two more simple filters. We would like to tress that here we take advantage of different factorization than in lift algorithm. This article represents a development of methods introduced in [12] and [13].

The rest of the letter is organized as follows: In section 2 we briefly describe the convolution based wavelet computation algorithm. In Sections 3 and 4 we focus on CDF and Daubechies and derive more computationally efficient algorithms for calculating CDF and Daubechies transforms. Finally, in the last two sections we provide the comparative evaluations and give conclusions.

2. WAVELET TRANSFORM - PYRAMIDAL ALGORITHM

In the pyramidal algorithm, a single level of resolution of the discrete wavelet transform (DWT) is calculated as follows: The input signal $x(n)$ is filtered with two mirror filters: $H_p(\omega)$ and $G_p(\omega)$, where $H_p(\omega)$ is a low-pass filter, $G_p(\omega)$ is a high-pass filter, p is a wavelet order and n=0,1,... [11]. The output U_1 of the low-pass filter $H(z)$ is called approximation (or coarse scale) and the high-frequency output Y_1 from $G(z)$ is called detail. The outputs of both filters are decimated. The above-mentioned procedure constitutes a single step of DWT. The following levels of resolution are calculated recursively, with the approximation signal U_n taken as an input signal, n=0,1,... . The structure of the DWT pyramidal algorithm is illustrated in fig. 1.

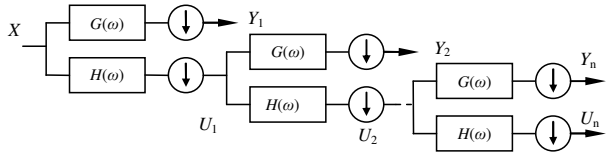

Fig. 1. The generalized structure of the DWT pyramidal algorithm

In [14] Daubechies formulated a condition, imposed on $H_p(\omega)$ and $G_p(\omega)$, which assures perfect reconstruction of compactly supported orthogonal and bi-orthogonal wavelets (1):

$$H_p(\omega)\cdot\overline{G_p(\omega+\pi)}=2\left(\frac{1+\cos\omega}{2}\right)^p\sum_{k=0}^{p-1}\binom{p+k-1}{k}\left(\frac{1-\cos\omega}{2}\right)^k, \quad (1)$$

where:
$H_p(\omega)$ - power frequency response low-pass filter,
$G_p(\omega)$ - power frequency response high-pass filter,
p - the coefficient which depends on wavelet order,
ω - frequency,
$\overline{G_p(\omega+\pi)}$ - complex conjugate of $G_p(\omega+\pi)$.

Taking into consideration the condition (1) and the fact that Daubechies filters are orthogonal ($H(\omega)=G(\omega+\pi)$) the filters $H(\omega)$ and $G(\omega)$ can be expressed in the following form in Z domain [14]:

$$H_p(z)=\delta_p\left(1+z^{-1}\right)^p\prod_{k=1}^{p-1}\left(1-\lambda_k z^{-1}\right) \quad (2)$$

$$G_p(z)=\delta_p\left(1-z^{-1}\right)^p\prod_{k=1}^{p-1}\left(1+\lambda_k z^{-1}\right) \quad (3)$$

where:
$H_p(z)$ - low-pass wavelet filter in Z - domain,
$G_p(z)$ - high-pass wavelet filter in Z - domain,
δ_p - scaling factor,
λ_p - coefficient.

To find the value of the coefficient λ_k we must first calculate the roots x_k of the polynomial given by (4) [14]:

$$B_p(x)=\sum_{n=0}^{p-1}\binom{p+n-1}{n}x^n \quad (4)$$

Next, we calculate λ_k from:

$$\lambda_k=\begin{cases}1-2x_k+\sqrt{(1-2x_k)^2-1} & when\ \ 1-2x_k+\sqrt{(1-2x_k)^2-1}\le 1,\\ 1-2x_k-\sqrt{(1-2x_k)^2-1} & when\ \ 1-2x_k+\sqrt{(1-2x_k)^2-1}>1.\end{cases} \quad (5)$$

CDF filters follow the similar pattern as Daubechies filters, but filter coefficients are computed in a different manner. In CDF wavelet transforms the condition given by (1) can be transformed to (6) in Z domain:

$$\begin{aligned}&H_p(z)\cdot G_p\left(-z^{-1}\right)=\\&=2\left(\frac{1}{4}z^{-1}\right)^p(1+z)^{2p}\sum_{k=0}^{p-1}\binom{p+k-1}{k}\left(-\frac{1}{4}z+\frac{1}{2}-\frac{1}{4}z^{-1}\right)^k.\end{aligned} \quad (6)$$

3. FACTORED FORM OF DAUBECHIES WAVELET FILTERS

In this section we transform the wavelet filters given by (2) and (3) to derive the more computationally efficient form of pyramidal algorithm for calculating Daubechies wavelet transform. The filters (2) and (3) can be expressed in the following, equivalent form respectively:

$$H_p(z)=\sum_{k=0}^{p}\binom{p}{k}z^{-k}\cdot\sum_{k=1}^{p}m_{k-1}(-1)^{k-1}z^{-k+1} \quad (7)$$

$$G_p(z)=\sum_{k=0}^{p}\binom{p}{k}(-1)^k z^{-k}\cdot\sum_{k=1}^{p}m_{p-k}z^{-k+1} \quad (8)$$

where the value m_i for i = 1,2,... can be found from (9):

$$\begin{cases}m_1=\sum_{k=1}^{p-1}\lambda_k, m_2=\sum_{k_1=1}^{p-2}\sum_{k_2=k_1+1}^{p-1}\lambda_{k_1}\lambda_{k_2}, m_3=\sum_{k_1=1}^{p-3}\sum_{k_2=k_1+1}^{p-2}\sum_{k_3=k_2+1}^{p-1}\lambda_{k_1}\lambda_{k_2}\lambda_{k_3},\\ m_{p-1}=\sum_{k_1=1}^{1}\sum_{k_2=2}^{2}\cdots\sum_{k_{p-2}=k_{p-3}+1}^{p-2}\sum_{k_{p-1}=k_{p-2}+1}^{p-1}\lambda_{k_1}\lambda_{k_2}\ldots\lambda_{k_{p-2}}\lambda_{k_{p-1}}.\end{cases} \quad (9)$$

Notice, that (7) and (8) are almost identical, the only difference is the $(-1)^k$ factor. Thus, the values of positive and negative coefficients in (7) and (8) can be grouped together, which leads to (10) and (11):

$$H_p(z)=\left(A_p(z)+B_p(z)\right)\cdot C_p(z), \quad (10)$$

$$G_p(z)=\left(A_p(z)-B_p(z)\right)\cdot D_p(z), \quad (11)$$

where:

$$A_p(z)=\sum_{k=0}^{p/2}\binom{p}{2k}z^{-2k},\ B_p(z)=\sum_{k=0}^{p/2}\binom{p}{2k+1}z^{-2k-1},$$

$$C_p(z)=\sum_{k=1}^{p}m_{k-1}(-1)^{k-1}z^{-k+1},\ D_p(z)=\sum_{k=1}^{p}m_{p-k}z^{-k+1}.$$

The wavelet transform algorithm structure corresponding to (10) and (11), is depicted in fig.2. The algorithm for calculating the wavelet transform from (10) and (11), is given as follows: First, even and odd samples of the input signal $x(n)$ are separated and filtered with two filters respectively: $A(z)$ and $B(z)$. Next, the outputs of both filters are added and subtracted. The resultant two signals are convolved with filters $C(z)$ and $D(z)$ respectively. Notice, that impulse responses of $A(z)$ and $B(z)$ are integers, so only integer multiplications must be computed to get the outputs of $A(z)$ and $B(z)$ filters.

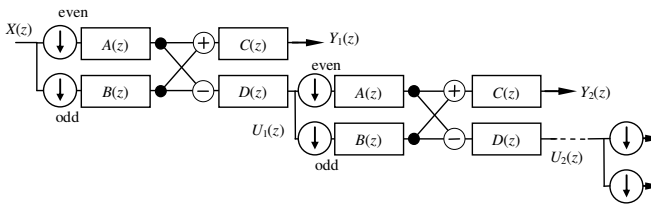

Fig. 2. The structure of the new algorithm for computing Daubechies transforms

4. FACTORED FORM OF CDF FILTERS

The condition (1) can be transformed into (12), which leads to the family of CDF wavelets:

$$H_p(z)\cdot G_p(-z^{-1}) = 2\left(\frac{1}{4}z^{-1}\right)^p (1+z)^{2p}\prod_{k=0}^{K}\left(\left(-\frac{1}{4}z+\frac{1}{2}-\frac{1}{4}z^{-1}\right)-x_k\right)+ \prod_{i=0}^{I/2}\left(\left(-\frac{1}{4}z+\frac{1}{2}-\frac{1}{4}z^{-1}\right)^2 - m_{2i-1}\left(-\frac{1}{4}z+\frac{1}{2}-\frac{1}{4}z^{-1}\right)+m_{2i}\right), \quad (12)$$

where:
$K=(l_{H0}-1)/2$,
$I=p-K-1$,
l_H - length of the $H_p(z)$ filter,
x_k (x_i)- k-th (i-th) root of the polynomial (4),
$m_{2i}=x_i x_{i+1}$,
$m_{2i-1}=x_i - x_{i+1}$.

CDF wavelet transform algorithm has very similar structure to Daubechies wavelet transform algorithm and can be calculated using similar algorithm. However, some changes must be made to the filter's structure, because, unlike in Daubechies wavelet transforms, in CDF wavelet transforms, the low frequency filter length may differ from high frequency filter length. Besides both filters have odd number of coefficients. As a result CDF wavelet filters cannot be transformed in the same manner as Daubechies filters. As an example let us discuss CDF 5/3 wavelet filter pair, which has the following coefficients $H(z) = [h_0, h_1, h_2, h_3, h_4]$ and $G(z) = [g_0, g_1, g_2]$. The algorithm is created in the following manner: First we take the common part of both filters: that is middle coefficients of low-pass filter: h_1, h_2, h_3 and all coefficients of high-pass filter: g_0, g_1, g_2. Using the common part of those filters build the efficient algorithm as described in section 3. Next, we add extra coefficient h_0 and h_4.

In general case the coefficients a_1, b_1, c_1, a_2, b_2, c_2, ... from fig. 3 must be computed separately for each output from $A_p^1(z)$, $B_p^1(z)$, $C_p^1(z)$, $A_p^2(z)$, $B_p^2(z)$, $C_p^2(z)$, ... and they can not be grouped together to become a single filter (see fig. 3). But the common part of all filters $A_p^1(z)$, $B_p^1(z)$, $C_p^1(z)$, $A_p^2(z)$, $B_p^2(z)$, $C_p^2(z)$, can be reduced in the same way as described in section 3.

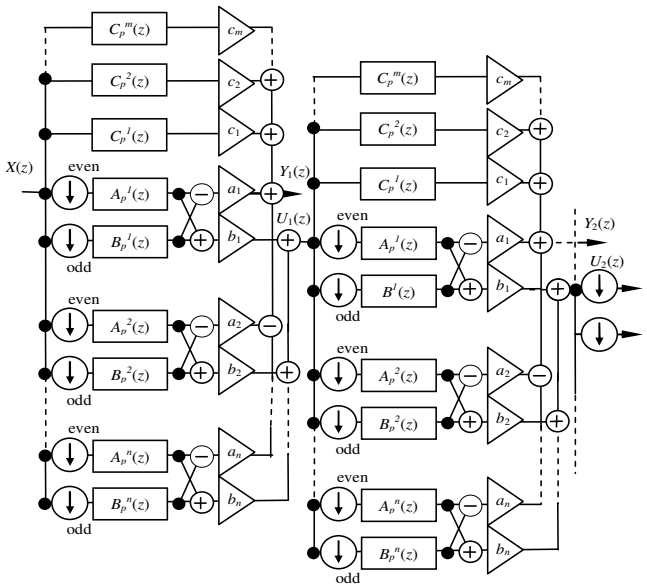

Fig. 3. The structure of the modified pyramidal algorithm for CDF filters

5. COMPUTATIONAL COMPLEXITY

In this section we compare the number of arithmetical operations required to compute wavelet transforms taking advantage of three algorithms: convolution-based, lift and the new algorithm introduced here. Here we examine the number of additions/subtractions, multiplications by 2 and multiplications required to calculate a single step of Daubechies and CDF wavelet transforms. We assume that calculating a single step is equivalent to calculating two output samples of a wavelet transform from two input samples of the input sequence.

The comparison between convolution-based, lift and the new algorithms for Daubechies and 9/7 CDF is given in table 1.

Table 1. Number of additions/subtractions (a/s), multiplication by 2 (2mul) and multiplications (mul) required to calculate a single step of wavelet transform for the convolution-based, the lift and the new algorithm.

operation	a/s	2mul	mul
	Convolution based algorithm		
Daubechies 4	6	0	8
Daubechies 6	10	0	12
CDF 9/7	14	0	16
	Lift algorithm		
Daubechies 4	4	0	5
Daubechies 6	6	0	8
CDF 9/7	8	0	6
	The new algorithm		
Daubechies 4	12	6	2
Daubechies 6	11	3	4
CDF 9/7	6	18	3

Notice that the new algorithm requires fewer multiplications than convolution-based and lift algorithms. The total number of arithmetical operations required to calculate a single step of Daubechies and CDF wavelet transforms using the new algorithm is higher, but resource consuming operations (multiplications) were replaced by less resource consuming additions/subtractions and multiplications by 2.

If we compare the number of clock cycles required to calculate a single step of wavelet transform when the algorithm is implemented in DSP or number of gates when implemented in VLSI , the MPWA requires fewer clock cycles as well as fewer gates than lift and convolution-based algorithms. For example, let us assume that:

– calculations are performed using 16-bit fixed point values,
– addition/subtraction requires 1 clock cycle
– multiplications require 10 clock cycles. This assumption is true when we consider 16-bit integer wavelet transform.
– each multiplication by 2 requires 1 clock cycle,

the new algorithm requires up to 68% fewer clock cycles than convolution-based one (see table 2).

Table 2. Number resources required to calculate a single step of wavelet transform for PWA and MPWA.

Transform	Convolution-based	lift	New algorithm
Daubechies 4	86	54	38
Daubechies 6	130	86	54
CDF 9/7	174	68	54

To verify the theoretical estimations from table 2, we have computed three sample transforms: Daubechies 4, 6 and CDF 9/7 consisting of a 16-bit, fixed point, random dataset (white noise) using convolution-based, lift and the new algorithm, in order to compare the time complexity of the algorithms. The comparison was performed on MC68EC030 microprocessor. The results obtained in practice, are very close to the theoretical estimations. Max difference from theoretical estimations varied below 5%.

6. CONCLUSIONS

In this article we have introduced the new algorithms which are more efficient than convolution based and lift algorithms when applied to CDF and Daubechies wavelet transforms. We have demonstrated that the new algorithms lead to resource saving effect. If we assume 16-bit integer transform the new algorithm requires up to 47% less resources than lift algorithm and up to 68% less resources than convolution-based algorithm.

7. REFERENCES

[1] Graps, A., "An Introduction to Wavelets", **IEEE Computational Science and Engineering**, Vol. 2. No. 2, 1995.

[2] Antonini, M., Barlaud, M., Mathieu, P., Daubechies, I., "Image coding using wavelet transform", **IEEE Trans. on Image Processing**, Vol. 1, No. 2, pp 205–220, 1992.

[3] Uzun, I.S., Amira, A., "Real-time 2-D wavelet transform implementation for HDTV compression", **IEEE Real Time Imaging,** Vol. 11, No. 2, pp 151-165, 2005.

[4] Daubechies I., Sweldens W., Factoring wavelet transforms into lifting steps. **Technical report, Bell Laboratories**, Lucent Technologies, 1996.

[5] Olkkonen H., Olkkonen J. T., Pesola P., Efficient Lifting Wavelet Transform for Microprocessor and VLSI Applications, **IEEE Signal Processing Letters**, Vol. 12 No. 2, February 2005.

[6] Akansu, A. N., "Multiplierless PR quadrature mirror filters for subband image coding", **IEEE Transactions on Image Processing**, vol. 5, pp 1359–1363, 1996.

[7] Kovacevic J. Sweldens W., "Wavelet Families of Increasing Order in Arbitrary Dimensions", **IEEE Transactions on Image Processing**, vol. 9, nr. 3, pp 480-496, 2000.

[8] Feil, M., Kutil, R., Meerwald, P., Uhl, A., "Wavelet Image and Video Coding on Parallel Architectures", **Proceedings of the 2nd IEEE Region 8 - EURASIP Symposium on Image and Signal Processing and Analysis** 2001.

[9] Grzeszczak, A., Mandal, M., K., Panchanathan, S., VLSI Implementation of Discrete Wavelet Transform, **IEEE Transactions on VLSI Systems**, Vol. 4, No. 4, pp 421-433, 1996.

[10] Cooklev T., An Efficient Architecture for Orthogonal Wavelet Transforms, **IEEE Signal Processing** Letters, Vol. 13, No 2, February 2006.

[11] Daubechies, I, "Ten Lectures on Wavelets", **CBMS-NSF Regional Conf. Series in Appl. Math**., Vol. 61. Society for Industrial nad Applied Mathematics. Philadelphia, PA, 1992.

[12] Lipiński, P., "Optimized 1-D Daubechies 6 Wavelet Transform", **Information Technologies and Systems**, Vol. 6, No 1-2, pp. 94-98, 2003.
[13] Lipiński, P., "Image Segmentation Method Using Wavelet Transform", **Proceedings of the VIITH International Conference CADSM**, Lviv – Slavske, Ukraine, pp. 495 - 496, 2003.
[14] Daubechies, I, "Orthonormal bases of compactly supported wavelets", **Commun on Pure and Applied Math**., 41, pp 909-996, 1988.