Chapter 1

IMAGE COMPRESSION USING SPLINE BASED WAVELET TRANSFORMS

Amir Z. Averbuch
School of Computer Science, Tel Aviv University
Tel Aviv 69978, Israel
amir@math.tau.ac.il

Valery A. Zheludev
School of Computer Science, Tel Aviv University
Tel Aviv 69978, Israel
zhel@math.tau.ac.il

Abstract In paper we describe a successful applications of the wavelet transforms to still image compression. The wavelet transforms were designed by the usage of discrete interpolatory splines. These filters outperform the traditional biorthogonal 9/7 filters which are frequenty used in wavelet based compression. The new filters and the biorthogonal 9/7 are incorporated into SPIHT in order to measure and compare their performance with one well known codec.

Keywords: Sample, edited book

Introduction

The three fundamental building blocks of compression systems, are transformation (such as Discrete Cosine Transform, wavelets), quantization (SQ, UTQ, etc.), and symbol modeling and encoding (Huffman, and arithmetic). In this paper we present new wavelet based filters which have good performance for still image compression and outperforms the traditional biorthogonal 9/7 filters which are frequenty used in wavelet based compression. The new filters and the biorthogonal 9/7 are incorporated into SPIHT [34] in order to measure and compare their performance with one well known codec.

0.1. Wavelet Based Image Coders

Wavelet transforms provide very good energy compaction: the transform decreases the correlation between the transformed coefficients. Even though the correlation between wavelet coefficients across scale is very small, the coefficients are not independent (there is no contradiction, since the probability density function of wavelet coefficients of natural image are not Gaussian). In fact, a visual inspection of the wavelet coefficients of an image will reveal that there are still coherent structures in the higher frequency bands. Furthermore these structures have a self similar structure across the different subbands.

While the wavelet coders are based on the wavelet decomposition and its multiresolution, the JPEG is based on 8×8 windowed Fourier transform. Therefore, JPEG ignores correlations among pixels over larger areas. This causes "blocking" effect in deep compression. Also, while the DCT-based image coders perform very well at moderate bit rates, at higher compression ratios image quality degrades because of the artifacts resulting from the block-based DCT scheme. Wavelet-based coding on the other hand provides substantial improvement in picture quality at low bit rates. Because of the inherent multiresolution nature, wavelet-based coders facilitate progressive transmission of images thereby allowing variable bit rates.

Over the past few years, a variety of novel and sophisticated wavelet-based image coding schemes have been developed. A family of algorithms, known as zerotree coders, exploit both the inter-band self-similarity, as well as the intraband coherent structure, and the dependencies across subbands. These include wavelet codecs ([2, 5, 6, 24, 3]), Embedded Zero Tree Wavelet (EZW) [36], Set Partitioning in Hierarchical Trees (SPIHT) [34], which uses the 9-7 biorthogonal filters [21], instead of the 9 tap filters of [1]), Space-Frequency Quantization for Wavelet Image Coding (SFQ) [44], which addresses the problem of how spatial quantization modes and standard scalar quantization can be applied in a jointly optimal fashion in an image coder, Efficient Pre-Coding Techniques for Wavelet-Based Image Compression (PACC) [29], introduces a coding method using a fast wavelet transform and an uniform quantizer combined with a framework of preceding techniques which are based on the concepts of partitioning, aggregation and conditional coding- PACC. Following these concepts, the data object emerging from the quantizer is first partitioned into different subsources. Parts of correlations within and between different subsources are then captured by aggregating homogeneous elements into data structures like run-length codes or zerotrees), EQ[28], (Image Coding Based on Mixture Modeling of Wavelet Coefficients and a Fast Estimation-Quantization Framework introduces an image compression paradigm that combines compression efficiency with speed, and is based on an independent "infinite" mixture model which accurately captures the space-frequency characterization of the wavelet image representation), Morphological Representation of Wavelet Data (MRWD) [35], (presents both an experimental study of the statistics of wavelet data, as well as the design of two different morphology-based coding algorithms, that make use of these statistics), SLCCA[12], (Significance-Linked Connected Component Analysis for Wavelet Image Coding, is a wavelet image coder which extends MRWD by exploiting both within-subband clustering of significant coefficients and cross-subband dependency in significant fields), Context Based (C/B)[16], (Context-Based Entropy Coding for Lossy Wavelet Image Compression which is an adaptive image coding algorithm based on backward adaptive quantization-classification techniques using a simple uniform scalar quantizer to quantize the image subbands), OC [26], Optimal Classification in Subband Coding of Images investigates various classification techniques, applied to subband coding of images, as a way of exploiting the non-stationary nature of image subbands), CREW[13], EPWIC[14], EBCOT[38], (Scalable Image Compression which is based on independent Embedded Block Coding with Optimized Truncation of the embedded bit-streams, which identifies some of the major contributions of the algorithm. The EBCOT algorithm [38] uses a wavelet transform to generate the subband coefficients which are then quantized and coded. Although the usual dyadic wavelet decomposition is typical, other "packet" decompositions are also supported and occasionally preferable), SR([39]), Image Coding using Adaptive Wavelets[41] (the wavelet filter should be chosen adaptively depending on the statistical nature of image being coded), Second Generation Image Coding[24], Image Coding using Wavelet Packets ([4, 19]), (larger libraries of waveforms which have been developed in order to describe long oscillatory patterns. The selected collection of patterns is called the "best basis". It is demonstrated that, despite this difficulty, the freedom to choose an adapted basis remains an enormous advantage), Wavelet Image Coding using VQ ([3]), and Lossless Image Compression using Integer Lifting [15], and hybrid codec that combine wavelet and waveletpacket [10] or any other combination.

The emerging standard, called JPEG-2000 [46], is being developed in two parts and is based upon wavelet decomposition. Combined with powerful quantization and encoding strategies such as embedded quantization and context based arithmetic coding, the use of wavelets in JPEG-2000 provides the potential for numerous advantages over the existing JPEG standard. Performance gains include improved compression efficiency at low bit rates or for large images, while new functionalities include multi-resolution representation, SNR scalability and embedded bit stream architecture, lossy to lossless progression, region-of-interest (ROI) coding, and a rich file format, random access and processing of separate parts of picture, robustness to bit errors, open architecture, content based description and interface with MPEG-4 [47].

0.2. Usage of splines for wavelet design

By now two ways were pursued for the construction of wavelet schemes via the usage of splines. One is to construct orthogonal and semi-orthogonal wavelets in the spline spaces (Battle-Lemarié [11, 27], Chui-Wang [17], Unser-Aldroubi-Eden [40]), Zheludev [45]. Another way was introduced by Cohen, Daubechies and Feauveau [20] who constructed symmetric compactly supported spline wavelets whose duals, remaining compactly supported and symmetric, do not belong to a spline space. However, since the introduction of the lifting scheme for the design of wavelet transforms [37], a new way was opened to use splines as a tool for devising a full discrete scheme of wavelet transforms.

The basic lifting scheme for wavelet transform of a discrete -time signal $\mathbf{x}$ consists of three steps:

Split – The signal is split into even and odd subarrays: $\mathbf{s} = \{s(k) = x(2k)\}, \quad \mathbf{d} = \{d(k) = x(2k+1)\}, \ k \in \mathbb{Z}$.

Predict - Some linear combinations of terms of the even array $\mathbf{s}$ are used to predict the odd array $\mathbf{d}$. Then, the array $\mathbf{d}$ is redefined as the difference between the existing array and the predicted one. If the predictor is chosen correctly, this step decorrelates the signal and reveals its high-frequency component.

Update – To eliminate aliasing, which appears while downsampling the original signal $\mathbf{x}$ into $\mathbf{s}$ and, by this means, to obtain the low-frequency component of the signal, the even array is updated using the new odd array.

The newly produced even and odd subarrays are the coefficients of one decomposition step of a wavelet transform $\mathbf{s}$ (low-frequency) and $\mathbf{d}$ (high-frequency). The inverse transform is implemented in a reverse order. The transform generates biorthogonal wavelet bases for the signal space. The specifics of the transform and its generated wavelets are determined by the choice of the predicting and updating aggregates.

In the construction by Donoho [23], which later was modified by Sweldens [37], an odd sample is predicted from a polynomial interpolation of neighboring even samples. Wavelets, which were generated by these transforms, are symmetric and compactly supported. Since the transform is interpolating, it operates immediately on the samples of the signal. However, these wavelet transforms are not efficient in applications.

New opportunities for design of wavelet transforms become available by usage of splines instead of polynomials as the predicting and updating aggregates in the lifting schemes.

Continuous interpolatory splines. The interpolatory spline of odd order $2m-1$ (even degree) with equidistant nodes possesses a remarkable property of super-convergence in the midpoints of the intervals between grid points k/N [43]. In these points it approximates the smooth function f with an accuracy of N^{-2m} whereas the global approximation accuracy is $N^{-(2m-1)}$. Thus we build the spline of an odd order $2m-1$ which interpolates the even samples of the signal $\mathbf{x}$ and predict the odd samples by the values of the spline in the midpoints of the intervals between the grid points. The prediction is exact on the polynomials of degrees up to $2m$. This leads to the decomposition wavelets with $2m+1$ vanishing moments. To supply the reconstruction wavelets with similar property, the even array can be updated by adding the values in the midpoints between the grid points of the spline, which interpolates the new odd samples (divided by two). The order of the update spline may differ from the order of the spline, which was employed for prediction. This scheme is described in our paper [9].

Discrete interpolatory splines. Another option is to use the discrete rather than continuous interpolatory splines. We describe the discrete splines construction in [7, 8]. In this case explicit formulas for the transforms with any number of vanishing moments are established. Moreover, our investigation revealed an interesting relation between the discrete splines and the Butterworth filters commonly used in signal processing. The filter banks, which are used in our scheme, comprise filters which act as a bi-directional half-band Butterworth filters. The frequency response of Butterworth filters are maximally flat and we succeeded in construction of the dual filters with similar property. Unlike the construction in [23], the designed transforms are using causal and anti-causal linear phase filters with infinite impulse response

(IIR). However , the transfer functions of the employed filters are rational. Therefore, filtering can be performed in a recursive manner. We established explicit formulas which enable fast cascade or parallel implementation. The boundaries are handled using symmetric extensions of the signals. The one-pass Butterworth filters were used already for devising orthogonal non-symmetric wavelets [25]. The computations there were conducted in time domain using recursive filtering. A scheme using recursive filters for the construction of biorthogonal symmetric wavelets was presented in [32], [30].

In the present paper we describe application of the wavelet transforms designed with usage of the discrete interpolatory splines to image compression. Details of construction and proofs of formulated propositions can be found in [8]. The rest of the paper is organized as follows. In Section 1 we recall the notion and outline necessary properties of the z-transform, Butterworth filters and interpolatory discrete splines. In Section 2 we introduce a family of biorthogonal wavelet-type transforms of discrete-time signals, which we construct through lifting steps. In Section 3 we interpret devised scheme as a transformation of the signals by a filter bank that possesses the perfect reconstruction properties. We reveal relation of this filter bank to Butterworth filters. In Section 3.2 we discuss choosing the control filter which is used in the update step. In Section 4 we describe recursive implementation of the transforms. We present general formulas which allow fast cascade or parallel implementation of transforms of any order. Then we give examples of most practical filters. The perfect reconstruction filter banks, that were constructed in previous sections, are associated with the biorthogonal pairs of of wavelet-type bases in the space of discrete-time signals. We describe these bases in Section 5. In Section 6 we explain one multiscale advance of the devised wavelet transforms. Section 7 is devoted to the presenta of results of experiments on image compression using devised biorthogonal transforms. In Section 7.1 we describe in details the transforms, which we employed in the experiments. In Section 7.2 we discuss computational complexity of the transforms. In Section 7.3 we present results of compression of four well-known benchmark images.

1. Preliminaries

1.1. z-transform

The sequences $\{a(k)\}_{k=-\infty}^{\infty}$, which belong to the space l_1, we call the discrete-time signals. The space of discrete-time signals we denote by $\mathcal{S}$. The $z-$transform of a signal $\{a(k)\} \in \mathcal{S}$ is defined as follows:

$$a(z) = \sum_{k=-\infty}^{\infty} z^{-k}\, a(k).$$

Throughout the paper we assume that $z = e^{i\omega}$. We recall the following properties of the $z-$transform:

$$\begin{aligned}
&a(k) = \sum_{l=-\infty}^{\infty} b(k-l)c(l) \Longleftrightarrow a(z) = b(z)\, c(z)\\
&a_e(z^2) \triangleq \sum_{k=-\infty}^{\infty} z^{-2k}\, a(2k) = \frac{1}{2}\,(a(z) + a(-z))\\
&a_o(z^2) \triangleq \sum_{k=-\infty}^{\infty} z^{-2k}\, a(2k+1) = \frac{z}{2}\,(a(z) - a(-z))\\
&a(z) = a_e(z^2) + z^{-1} a_o(z^2).\\
&za(z) = \sum_{k=-\infty}^{\infty} z^{-k}\, a(k+1),
\end{aligned}$$

that is $za(z)$ is the $z-$transform of the shifted signal $\{a(k+1)\}$.

1.2. Discrete splines

The discrete B-spline of the first order is defined by the following sequence:

$$B_{1,n}(j) = \begin{cases} 1 & \text{if } j = 0, \ldots, 2n-1,\ n \in \mathbb{N}, \\ 0, & \text{otherwise}, \quad j \in \mathbb{Z}. \end{cases}$$

We define the higher order B-splines as the discrete convolutions by recurrence: $B_{p,n} = B_{1,n} * B_{p-1,n}$. Obviously, the z-transform of the B-spline of order p is

$$B_{p,n}(z) = (1 + z^{-1} + z^{-2} + \ldots + z^{-2n+1})^p \quad p = 1, 2, \ldots.$$

In this paper we are interested only in the case when $p = 2r,\ r \in \mathbb{N}$ and $n = 1$. The corresponding splines are denoted as $B_r = B_{2r,1}$. In this case we have $B_r(z) = (1 + z^{-1})^{2r}$. The B-spline $B_r(j)$ is symmetric about the point $j = r$ where it attains its maximal value. We define the central B-spline $Q_r(j)$ of order $2r$ as a shift of the B-spline:

$$Q_r(j) \triangleq B_r(j+r), \quad Q_r(z) = z^r B_r(z) = z^r (1 + z^{-1})^{2r}.$$

The discrete spline of order $2r$ is defined as a linear combination, with real-valued coefficients, of shifts of the central B-spline of order $2r$:

$$S_r(k) \triangleq \sum_{l=-\infty}^{\infty} c(l) Q_r(k-2l).$$

Definition 1.1 *Let $\{e(k)\} \in S$ be a given sequence. The discrete spline S_r is called the interpolatory spline if the following relations hold:*

$$S_r(2k) = e(k), \;\; k \in \mathbb{Z}. \qquad (1.1)$$

The points $\{2k\}$ are called the nodes of the spline.

The following proposition shows how interpolatory splines of any order can be constructed. Moreover, for further development we need to know the values of the splines in the midpoints between the nodes, which we denote as $\sigma(k) = S_r(2k+1), \; k \in \mathbb{Z}$.

Proposition 1.1 *The interpolatory spline which satisfies the conditions (1.1) is represented as follows*

$$S_r(k) = \sum_{l=-\infty}^{\infty} c(l) Q_r(k-2l), \quad c(z^2) = \frac{2e(z^2)}{z^r \left(1+z^{-1}\right)^{2r} + (-z)^r \left(1-z^{-1}\right)^{2r}}.$$

The $z-$transform of the interpolatory spline in the midpoints are

$$\sigma(z^2) = z U_r(z) e(z^2), \quad U_r(z) \triangleq \frac{\left(1+z^{-1}\right)^{2r} - (-1)^r \left(1-z^{-1}\right)^{2r}}{\left(1+z^{-1}\right)^{2r} + (-1)^r \left(1-z^{-1}\right)^{2r}}. \quad (1.2)$$

In addition, $U_r(-z) = -U_r(z)$.

1.3. Discrete-time Butterworth filters

We recall briefly the notion of Butterworth filter. For details we refer to [31]. The input $x(n)$ and the output $y(n)$ of a linear discrete time shift-invariant system are linked as

$$y(n) = \sum_{k=-\infty}^{\infty} f(k) x(n-k). \qquad (1.3)$$

Such a processing of the signal $x(n)$ is called digital filtering and the sequence $f(n)$ is called the impulse response of the filter. Its $z-$transform $f(z) = \sum_{n=-\infty}^{\infty} z^{-n} f(n)$ is called the transfer function of the filter. Denote by $X(\omega) = \sum_{n=-\infty}^{\infty} e^{-i\omega n} x(n)$, $Y(\omega) = \sum_{n=-\infty}^{\infty} e^{-i\omega n} y(n)$, $F(\omega) = \sum_{n=-\infty}^{\infty} e^{-i\omega n} f(n)$ the discrete Fourier transforms of the sequences. Then, we have from (1.3) $Y(\omega) = F(\omega)X(\omega)$. The function $F(\omega)$ is called the frequency response of the digital filter. The digital Butterworth filter is a filter with a maximally flat frequency response. The magnitude squared frequency responses $F_l(\omega)$ and $F_h(\omega)$ of the digital low-pass and high-pass Butterworth filters of order r, respectively, are given by the formulas

$$|F_l(\omega)|^2 = \frac{1}{1 + (\tan\frac{\omega}{2}/\tan\frac{\omega_c}{2})^{2r}},$$
$$|F_h(\omega)|^2 = 1 - |F_l(\omega)|^2 = \frac{1}{1 + (\tan\frac{\omega_c}{2}/\tan\frac{\omega}{2})^{2r}}$$

where ω_c is the cutoff frequency.

We are interested in the half-band Butterworth filters that is $\omega_c = \pi/2$. In this case

$$|F_l(\omega)|^2 = \frac{1}{1 + (\tan\frac{\omega}{2})^{2r}}, \quad |F_h(\omega)|^2 = 1 - |F_l(\omega)|^2 = \frac{1}{1 + (\cot\frac{\omega}{2})^{2r}}.$$

If we put $z = e^{i\omega}$ then we obtain the magnitude squared transfer function of the low-pass filter:

$$|f_l(z)|^2 = \frac{\left(1 + z^{-1}\right)^{2r}}{\left(1 + z^{-1}\right)^{2r} + (-1)^r\left(1 - z^{-1}\right)^{2r}} \tag{1.4}$$

Similarly, we have the magnitude squared transfer function of the high-pass filter:

$$|f_h(z)|^2 = \frac{(-1)^r\left(1 - z^{-1}\right)^{2r}}{\left(1 + z^{-1}\right)^{2r} + (-1)^r\left(1 - z^{-1}\right)^{2r}} \tag{1.5}$$

It is readily seen that the function U_r defined in (1.2), is related to these transfer functions:

$$1 + U_r(z) = 2|f_l(z)|^2, \quad 1 - U_r(z) = 2|f_h(z)|^2. \tag{1.6}$$

2. Biorthogonal transforms

We introduce a family of biorthogonal wavelet-type transforms that operate on the signal $\mathbf{x} = \{x(k)\}_{k=-\infty}^{\infty}$, which we construct through lifting steps. We carry out the construction in the $z-$ domain and discuss the time-domain implementation in subsequent sections.

The lifting scheme can be implemented in a primal or dual modes. We consider only the primal mode because this scheme steadily outperforms the dual one in image processing applications.

2.1. Decomposition

Generally, the primal lifting scheme for decomposition of signals consists of three steps: 1. Split. 2. Predict. 3. Update or lifting. Let us construct our proposed schemes in terms of these steps.

Split - We split the array $\mathbf{x}$ into an even and odd sub-arrays:

$$\mathbf{e_1} = \{e_1(k) = x(2k)\}, \quad \mathbf{d_1} = \{d_1(k) = x(2k+1)\}, \ k \in \mathbb{Z}.$$

Predict - We use the even array $\mathbf{e_1}$ to predict the odd array $\mathbf{d_1}$ and redefine the array $\mathbf{d_1}$ as the difference between the existing array and the predicted one. To be specific, we use the spline S_r which interpolates the sequence $\mathbf{e_1}$ and predict the function $d_1(z^2)$ which is the z^2-transform of $\mathbf{d_1}$. It is predicted by the function $\sigma(\mathbf{z})$ defined in (1.2). The $z-$transform of the new $d-$array is defined as follows:

$$d_1^u(z^2) = d_1(z^2) - zU_r(z)e_1(z^2). \tag{1.7}$$

From now on the superscript u means an *update* operation of the array.

Lifting - We update the even array using the new odd array:

$$e_1^u(z^2) = e_1(z^2) + \beta(z)z^{-1}\, d_1^u(z^2). \tag{1.8}$$

Generally, the goal of this step is to eliminate aliasing which appears while downsampling the original signal $\mathbf{x}$ into $\mathbf{e_1}$. By doing so we have that $\mathbf{e_1}$ is transformed into a low-pass filtered and downsampled replica of $\mathbf{x}$. In Section 3.2 we will discuss how to achieve this effect by a proper choice of the control filter β , but for now we only require the function $\beta(z)$ to be real-valued and obey the condition $\beta(-z) = -\beta(z)$, so the product $\beta(z)z^{-1}$ is a function of z^2.

2.2. Reconstruction

The reconstruction of the signal $\mathbf{x}$ from the arrays $\mathbf{e_1^u}$ and $\mathbf{d_1^u}$ is implemented in reverse order: 1. Undo Lifting. 2. Undo Predict. 3. Unsplit.

Undo Lifting - We restore the even array:

$$e_1(z^2) = e_1^u(z^2) - \beta(z)z^{-1}\, d_1^u(z^2). \tag{1.9}$$

Undo Predict - We restore the odd array:

$$d_1(z^2) = d_1^u(z^2) + zU_r(z)e_1(z^2). \tag{1.10}$$

Unsplit - The last step represents the standard restoration of the signal from its even and odd components. In the $z-$ domain it looks as:

$$x(z) = e_1(z^2) + z^{-1}d_1(z^2). \tag{1.11}$$

3. Filter banks

3.1. Relation to Butterworth filters

Lifting schemes, that were presented above, yield efficient algorithms for the implementation of the forward and backward transform of $\mathbf{x} \longleftrightarrow \mathbf{e_1^u} \cup \mathbf{d_1^u}$. But these operations can be interpreted as transformations of the signals by a filter bank that possesses the perfect reconstruction properties.

First we define two filter transfer functions

$$\Phi_{l,r}(z) \stackrel{\Delta}{=} (1 + U_r(z))/2, \quad \Phi_{h,r}(z) \stackrel{\Delta}{=} (1 - U_r(z))/2. \tag{1.12}$$

¿From (1.6) it is clear that the linear phase filter with the transfer function $\Phi_{l,r}(z)$ is equal to the magnitude squared transfer function of the discrete-time low-pass half-band Butterworth filter of order r. The linear phase filter with the transfer function $\Phi_{h,r}(z)$ is equal to the magnitude squared transfer function of the high-pass half-band Butterworth filter. It means that application of these filters on a signal is equivalent to two passes applications (causal and anti-causal) of the corresponding Butterworth filters. We call these filters the bi-directional Butterworth filters.

Define the filter functions

$$\tilde{g}^r(z) \stackrel{\Delta}{=} 2z^{-1}\Phi_{h,r}(z), \quad \tilde{h}_\beta^r(z) \stackrel{\Delta}{=} 1 + 2\beta(z)\Phi_{h,r}(z) = 1 + \beta(z)\, z\tilde{g}^r(z), \tag{1.13}$$

$$h^r(j) \stackrel{\Delta}{=} 2\Phi_{l,r}(z) \quad g_\beta^r(z) \stackrel{\Delta}{=} z^{-1}\left(1 - 2\beta(z)\Phi_{l,r}(z)\right) = z^{-1}\left(1 - \beta(z)\, h^r(z)\right). \tag{1.14}$$

We call $\tilde{h}_\beta^r(z)$ and $\tilde{g}^r(z)$ the transfer functions of the low-pass and high-pass primal decomposition filters, respectively. We call $h^r(z)$ and $g_\beta^r(j)$

the transfer functions of the low-pass and high-pass primal reconstruction filters, respectively. These four filters form a perfect reconstruction filter bank ([42]).

Theorem 1 *The decomposition and reconstruction formulas can be represented as follows:*

$$e_1^u(z^2) = \frac{1}{2}\Big(\overline{\tilde{h}_\beta^r(z)}\, x(z) + \overline{\tilde{h}_\beta^r(-z)}\, x(-z)\Big) \quad (1.15)$$

$$d_1^u(z^2) = \frac{1}{2}\Big(\overline{\tilde{g}^r(z)}\, x(z) + \overline{\tilde{g}^r(-z)}\, x(-z)\Big) \quad (1.16)$$

$$x(z) = h^r(z) e_1^u(z^2) + g_\beta^r(z)\, d_1^u(z^2). \quad (1.17)$$

If $\beta(-z) = -\beta(z)$ then the functions $\tilde{h}_\beta^r(j)$, $\tilde{g}^r(j)$, $h^r(j)$ and $g_\beta^r(j)$ satisfy the perfect reconstruction conditions

$$h^r(z)\,\overline{\tilde{h}_\beta^r(z)} + g_\beta^r(z)\,\overline{\tilde{g}^r(z)} = 2$$
$$h^r(z)\,\overline{\tilde{h}_\beta^r(-z)} + g_\beta^r(z)\,\overline{\tilde{g}^r(-z)} = 0. \quad (1.18)$$

The following is an obvious observation.

Proposition 3.1 *The filter functions are linked to each other in the following way:*

$$\tilde{g}^r(z) = z^{-1} h^r(-z); \quad g_\beta^r(z) = z^{-1}\,\tilde{h}_\beta^r(-z)$$

Remark . We notice that the reconstruction filter $h^r(z)$ is equal (up to constant factors) to the transfer function of the bi-directional low-pass half-band Butterworth filter of order r. The decomposition filter $\tilde{g}^r(z)$ multiplied by z is equal to the bi-directional high-pass half-band Butterworth filter.

3.2. Choosing the control filter

So far we did not specify how to choose the filter β, which occurs during the construction of the primal filters g_β^r and $\tilde{h}_\beta^r$ and the dual ones $\tilde{G}_\beta^R$ and H_β^R. The only imposed requirements were that the function $\beta(z)$ be real-valued and $\beta(-z) = -\beta(z)$. Therefore, we are free to use the function β_1 for custom design of these filters and the corresponding wavelets. We consider here only one approach how to choose the control filter $\beta(j)$ which results in retaining the maximal flatness of the filters.

As was mentioned above, the reconstruction filter h^r and the decomposition filter $\tilde{g}^r$ are equal to the bi-directional low- and high-pass half-band Butterworth filters of order r, $\Phi_{l,r}$ and $\Phi_{h,r}$, respectively. These filters are linear phase and maximally flat in their pass- and stop-bands due to the factors $(1+z^{-1})^{2r}$ for the low-pass filters and $(1-z^{-1})^{2r}$ for the high-pass filters (see(1.4) (1.5)). Moreover, since $\Phi_{h,r}(z)$ has root of order $2r$ at $z=1$, filtering with $\Phi_{h,r}$ eliminates discrete polynomials up to degree $2r-1$. Consequently, the corresponding analysis wavelets have $2r$ vanishing moments (we will discuss this in Section 5.) The low-pass transfer function linked to the high-pass one as follows: $\Phi_{l,r}(z) = 1 - \Phi_{h,r}(z)$. Thus, filtering with $\Phi_{l,r}$ restores discrete polynomials up to degree $2r-1$. We retain similar properties for filters which depend on β: $\tilde{h}^r_\beta$ and g^r_β. To achieve it, we choose for filters of order r:

$$\beta(z) = U_p(z)/2 = \Phi_{l,p}(z) - \frac{1}{2} = \frac{1}{2} - \Phi_{h,p}(z). \tag{1.19}$$

Proposition 3.2 *If filter β is chosen as in (1.19), where p is a natural number, then the analysis filter $\tilde{h}^r_\beta$ restores discrete polynomials up to degree $2r-1$ and the synthesis filter g^r_β eliminates discrete polynomials up to degree $2q-1$, where $q=\min\{p,r\}$.*

Proof: Consider first the analysis filter

$$\tilde{h}^r_\beta(z) = 1 + \frac{1}{2}U_p(z)\Big(1 - U_r(z)\Big) = 1 + \Phi_{h,r}(z) - 2\Phi_{h,p}(z)\Phi_{h,r}(z).$$

Since $\Phi_{h,r}$ eliminates polynomials up to degree $2r-1$, it is clear that $\tilde{h}^r_\beta$ restores such polynomials. Similarly

$$\begin{aligned} g^r_\beta(z) &= z^{-1}\left(1 - \frac{1}{2}U_p(z)\Big(1 + U_r(z)\Big)\right) \\ &= z^{-1}\left[\Phi_{h,r}(z) + 2\Phi_{h,p}(z) - 2\Phi_{h,p}(z)\Phi_{h,r}(z)\right]. \end{aligned}$$

■

We mark the special case when $p=r$. Then

$$\begin{aligned} \tilde{h}_\beta(z) &= \Phi_{l,r}(z)(1 + 2\Phi_{h,r}(z)) \\ g_\beta(z) &= z^{-1}\left[\Phi_{h,r}(z)(1 + 2\Phi_{l,r}(z))\right]. \end{aligned}$$

We see that, as in the case of the filters h^r and $\tilde{g}^r$, the filters $\tilde{h}^r_\beta$ and g^r_β are also mirrored replicas of each other. They differ from bi-directional Butterworth filters of order r by the term $2\Phi_{l,r}(z)\Phi_{h,r}(z)$ which affects only the central part of the frequency domain.

4. Recursive implementation of the transforms

4.1. General formulas

Once we have chosen the control filter β as in (1.19), the decomposition procedure is the following (see (1.7), (1.8)):

$$d_1^u(z^2) = d_1(z^2) - zU_r(z)e_1(z^2) \tag{1.20}$$

$$e_1^u(z^2) = e_1(z^2) + \frac{1}{2}z^{-1}U_p(z)\, d_1^u(z^2). \tag{1.21}$$

The transfer function $z^{-1}U_p(z)/2$ differs from $zU_p(z)/2$ only by the factor z^{-2} that is the impulse responses of the corresponding filters are identical up to a shift. Thus both operations of the decomposition are, in principle, identical. For the reconstruction the same operation are conducted in a reverse order.

Therefore, it is sufficient to describe implementation of filtering with the function $zU_r(z)$. From (1.2) one can see that the function $zU_r(z)$ depends actually on z^2 and we denote it as

$$F_r(z^2) \triangleq zU_r(z) = z\frac{(1+z)^{2r} - (-1)^r\,(z-1)^{2r}}{(1+z)^{2r} + (-1)^r\,(z-1)^{2r}}. \tag{1.22}$$

We introduce a few notation. We distinguish between two cases.

Odd case: Let $r = 2q+1$. Then we denote for $k = 1, \ldots, q$:

$$\alpha_k^r = \cot^2\frac{(q+k)\pi}{2r} < 1, \quad \gamma_k^r = \cot^2\frac{(2q+2k+1)\pi}{4r} < 1 \tag{1.23}$$

Even case: Let $r = 2q$. Then we denote for $k = 1, \ldots, q$:

$$\alpha_k^r = \cot^2\frac{(2q+2k-1)\pi}{4r} < 1, \quad \gamma_k^r = \cot^2\frac{(q+k)\pi}{2r} < 1. \tag{1.24}$$

Proposition 4.1 *If $r = 2q+1$ then the function $F_r(z)$ is represented as follows:*

$$\begin{aligned} F_r(z) &= \frac{\alpha_1^r\alpha_2^r\ldots\alpha_q^r(1+z)\prod_{k=1}^q(1+\gamma_k^r z^{-1})(1+\gamma_k^r z)}{2r\gamma_1^r\gamma_2^r\ldots\gamma_q^r\prod_{k=1}^q(1+\alpha_k^r z^{-1})(1+\alpha_k^r z)} \\ &= A_r(1+z)\prod_{k=1}^q R_r(z,k)\prod_{k=1}^q R_r(z^{-1},k) \ \textit{where} \qquad (1.25) \\ A_r &\triangleq \frac{\alpha_1^r\alpha_2^r\ldots\alpha_q^r}{2r\gamma_1^r\gamma_2^r\ldots\gamma_q^r}, \quad R_r(z,k) \triangleq \frac{1+\gamma_k^r z}{1+\alpha_k^r z}. \end{aligned}$$

If $r = 2q$ *then*

$$\begin{aligned}F_r(z) &= \frac{2r\alpha_1^r\alpha_2^r \dots \alpha_q^r(1+z)\prod_{k=1}^{q-1}(1+\gamma_k^r z^{-1})(1+\gamma_k^r z)}{\gamma_1^r\gamma_2^r \cdots \gamma_{q-1}^r \prod_{k=1}^{q}(1+\alpha_k^r z^{-1})(1+\alpha_k^r z)} \\ &= A_r(1+z)\prod_{k=1}^{q} R_r(z,k)\prod_{k=1}^{q} R_r(z^{-1},k), \text{ where} \qquad (1.26)\\ A_r &\triangleq \frac{2r\alpha_1^r\alpha_2^r\dots\alpha_q^r}{\gamma_1^r\gamma_2^r\cdots\gamma_{q-1}^r}, \quad R_r(z,q) \triangleq \frac{1}{1+\alpha_q^r z}, \; R_r(z,k) \triangleq \frac{1+\gamma_k^r z}{1+\alpha_k^r z},\\ k &= 1,\dots,q-1.\end{aligned}$$

To illustrate the implementation we consider the primal decomposition procedure. Since $zU(z) = F_r(z^2)$ then the decomposition formulas (1.20), and (1.21) are equivalent to the following

$$d_1^u(z) = d_1(z) - F_r(z)e_1(z) \qquad (1.27)$$

$$e_1^u(z) = e_1(z) + \frac{1}{2z}F_p(z)\, d_1^u(z). \qquad (1.28)$$

Equation (1.27) means that in order to obtain the detail array $\mathbf{d}_1^u$, we must process the even array $\mathbf{e}_1$ by the filter F_r with the transfer function $F_r(z)$ and extract the filtered array from the odd array $\mathbf{d}_1$. Equation (1.28) means that in order to obtain the smoothed array $\mathbf{e}_1^u$, we must process the detail array $\mathbf{d}_1^u$ by the filter Φ_r that has the transfer function $\Phi_r(z) = z^{-1}F_r(z)/2$ and add the filtered array to the even array $\mathbf{e}_1$. But the filter Φ_r differs from $F_r/2$ only by one-sample delay and it operates similarly.

Proposition 4.1 implies that either of even and odd order filters F_{2q} and F_{2q+1} can be split into a cascade of q elementary causal recursive filters, denoted by $\overrightarrow{R_r(k)}$, with the transfer function $R_r(z^{-1},k)$, q elementary anti-causal recursive filters, denoted by $\overleftarrow{R_r(k)}$, with the transfer function $R_r(z,k)$ and the FIR filter Q with the transfer function $1+z$. On the other hand, the filters can be decomposed into sums of elementary recursive filters. Such a decomposition also allows parallel implementation of the transform.

The causal, anti-causal and the FIR filters operate as follows:

$$\mathbf{y} = \overrightarrow{R_r(k)}\mathbf{x} \Longleftrightarrow y(l) = x(l) + \gamma_k^r x(l-1) - \alpha_k^r y(l-1), \qquad (1.29)$$

$$\mathbf{y} = \overleftarrow{R_r(k)}\mathbf{x} \Longleftrightarrow y(l) = x(l) + \gamma_k^r x(l+1) - \alpha_k^r y(l+1), \qquad (1.30)$$

$$\mathbf{y} = Q\mathbf{x} \Longleftrightarrow y(l) = x(l) + x(l+1). \qquad (1.31)$$

Since the reconstruction in the lifting scheme differs from the decomposition only by the order of operations, its implementation is completely explained above.

4.2. Examples of recursive filters

Now we present a few particular cases with filters of various orders.

$r=1$. This is the simplest case. We have $F_1(z)=(1+z)/2$. The filter F_1 is reduced to the FI R filter $Q/2$.

$r=2$. In this case $\alpha_1^2=3-2\sqrt{2}\approx 0.172$ and

$$F_2(z)=4\alpha_1^2\frac{1+z}{(1+\alpha_1^2z)(1+\alpha_1^2z^{-1})}.$$

The filter can be implemented with the following cascade:

$$\begin{aligned} x_0(k) &= 4\alpha_1^2(x(k)+x(k+1)) \\ x_1(k) &= x_0(k)-\alpha_1^2x_1(k-1) \end{aligned} \qquad y(k)=x_1(k)-\alpha_1^2y(k+1). \tag{1.32}$$

Another option stems from the following decomposition of the function $F_2(z)$:

$$F_2(z)=\frac{4\alpha_1^2}{1+\alpha_1^2}\left(\frac{1}{1+\alpha_1^2z^{-1}}+\frac{z}{1+\alpha_1^2z}\right).$$

Then the filter is implemented in parallel mode:

$$\begin{aligned} y_1(k) &= x(k)-\alpha_1^2y_1(k-1) \\ y_2(k) &= x(k+1)-\alpha_1^2y_2(k+1) \end{aligned} \qquad y=\frac{4\alpha_1^2}{1+\alpha_1^2}\left(y_1+y_2\right). \tag{1.33}$$

We note that elementary filters, which produce y_1 and y_2, are operating in opposite directions.

$r=3$. In this case $\gamma_1^3=7-4\sqrt{3}\approx 0.0718$, $\alpha_1^3=1/3$ and

$$F_3(z)=\frac{1}{18\gamma_1^3}\cdot\frac{1+\gamma_1^3z}{1+z/3}\cdot\frac{1+\gamma_1^3z^{-1}}{1+z^{-1}/3}\cdot(1+z). \tag{1.34}$$

This formula leads to a cascade implementation of the transform. For the parallel implementation we use the following decomposition of the transfer function:

$$F_3(z)=\frac{1}{6}\left[-\frac{8}{1+z/3}-\frac{8}{9}\frac{z^{-1}}{1+z^{-1}/3}+z+\frac{35}{3}\right]. \tag{1.35}$$

$r=4$. Now $\gamma_1^4=3-2\sqrt{2}\approx 0.1716$, $\alpha_1^4=7-4\sqrt{2}-2\sqrt{20-14\sqrt{2}}\approx$ 0.4465,

$\alpha_2^4 = 7+4\sqrt{2} - 2\sqrt{20-14\sqrt{2}} \approx 0.0396$. So, we have the cascade representation:

$$F_4(z) = \frac{8\alpha_1^4\alpha_2^4}{\gamma_1^4} \cdot \frac{1+\gamma_1^4 z}{1+\alpha_1^4 z} \cdot \frac{1+\gamma_1^4 z^{-1}}{1+\alpha_1^4 z^{-1}} \cdot \frac{1}{1+\alpha_2^4 z} \cdot \frac{1}{1+\alpha_2^4 z^{-1}} \cdot (1+z).$$

Denote $r_1 = -20.751762964438$, $r_2 = -0.226770684430$, $r_3 = -0.020180988365$, $r_4 = -0.001285362767$. Then, the transfer function is represented as follows:

$$F_4(z) = 8\left[\frac{r_1\alpha_2^4}{1+\alpha_2^4 z} + \frac{r_2\alpha_1^4}{1+\alpha_1^4 z} + \frac{r_3 z^{-1}}{1+\alpha_1^4 z^{-1}} + \frac{r_4 z^{-1}}{1+\alpha_2^4 z^{-1}} + 1\right].$$

5. Bases for the signal space

The perfect reconstruction filter banks, that were constructed above, are associated with the biorthogonal pairs of bases in the space $\mathcal{S}$ of discrete-time signals.

In section 3 we introduced a family of filters by their transfer functions $h(z)$, $g_\beta^r(z)$, $\tilde{h}_\beta^r(z)$, $\tilde{g}^r(z)$. We denote by $\varphi^{r,1}(k)$, $\psi_\beta^{r,1}(k)$, $\tilde{\varphi}_\beta^{r,1}(k)$, $\tilde{\psi}^{r,1}(k)$ the impulse response functions of the corresponding filters, respectively. It means that, for example

$$h^r(z) = \sum_{k\in\mathbb{Z}} z^{-k}\varphi^{r,1}(k)$$

and similarly for the other functions.

Theorem 2 *The shifts of the functions $\varphi^{r,1}(k)$, $\psi_\beta^{r,1}(k)$, $\tilde{\varphi}_\beta^{r,1}(k)$ and $\tilde{\psi}^{r,1}(k)$ form a biorthogonal pair of bases for the signal space $\mathcal{S}$. This means that any signal $\mathbf{x} \in \mathcal{S}$ can be represented as:*

$$x(l) = \sum_{k\in\mathbb{Z}} e_1^u(k)\,\varphi^{r,1}(l-2k) + \sum_{k\in\mathbb{Z}} d_1^u\,\psi_\beta^{r,1}(l-2k).$$

The coordinates $e_1^u(k)$ and $d_1^u(k)$ can be represented as inner products:

$$e_1^u(k) = \langle \mathbf{x}, \tilde{\varphi}_{\beta,k}^1\rangle, \quad \textit{where } \tilde{\varphi}_{\beta,k}^1(l) = \tilde{\varphi}_\beta^{r,1}(l-2k) \qquad (1.36)$$

$$d_1^u(k) = \langle \mathbf{x}, \tilde{\psi}_k^{r,1}\rangle, \quad \textit{where } \tilde{\psi}_k^{r,1}(l) = \tilde{\psi}^{r,1}(l-2k). \qquad (1.37)$$

The following biorthogonal relations hold:

$$\langle \tilde{\varphi}_{\beta,k}^{r,1}, \varphi_l^{r,1}\rangle = \langle \psi_{\beta,k}^{r,1}, \tilde{\psi}_l^{r,1}\rangle = \delta_k^l, \quad \langle \tilde{\varphi}_{\beta,k}^{r,1}, \psi_{\beta,l}^{r,1}\rangle = \langle \tilde{\psi}_l^{r,1}, \varphi_k^{r,1}\rangle = 0, \quad \forall l,\ k.$$

The formulated theorem justifies the following definition.

Definition 5.1 *The functions $\varphi^{r,1}$ and $\psi_\beta^{r,1}$, which belong to the space $\mathcal{S}$, are called the low-frequency and high-frequency synthesis wavelets of the first scale, respectively. The functions $\tilde{\varphi}_\beta^{r,1}$ and $\tilde{\psi}^{r,1}$, which belong to the space $\mathcal{S}$, are called the primal low-frequency and high-frequency analysis wavelets of the first scale, respectively.*

Definition 5.2 *We say that a wavelet ψ has m vanishing moments if the following relations hold*

$$\sum_{k\in\mathbb{Z}} k^s\psi(k) = 0, \quad s = 0, 1, \ldots, m-1.$$

Proposition 5.1 *The high-frequency analysis wavelets of order r $\tilde{\psi}^{r,1}$ have $2r$ vanishing moments. If the control filter β is chosen as in (1.19), i.e. $\beta(z) = U_p(z)/2$, then the synthesis wavelet $\psi_b^{r,1}$ has $2q$ vanishing moments , where $q = \min\{p, r\}$.*

6. Multiscale wavelet transforms

Repeated applications of the transform can be achieved in an iterative way. It can be implemented by either a linear invertible transform of a wavelet type or by a wavelet packet type transform which results in an overcomplete representation of the signal. In this section we explain one multiscale advance of the wavelet transform.

In this transform we store the array $\mathbf{d}_1^u$ and decompose the array $\mathbf{e}_1^u$. The transformed arrays $\mathbf{e}_2^u$ and $\mathbf{d}_2^u$ of the second decomposition scale are derived from the even and odd sub-arrays of the array $\mathbf{e}_1^u$ by the same lifting steps as those described in Section 2. The transform is implemented using the recursive filters presented in Section 4. As a result we get that the signal $\mathbf{x}$ is transformed into three subarrays: $\mathbf{x} \leftrightarrow \mathbf{d}_1^u \bigcup \mathbf{d}_2^u \bigcup \mathbf{e}_2^u$. The reconstruction is performed in the reverse order.

Again, the transform leads to expansions of the signal with the biorthogonal pair of bases. As in Section 5, we present the $z-$transform of the signal as follows:

$$x(z) = x_h(z) + x_g(z) \text{ where } x_h(z) = h^r(z)e_1^u(z^2), \;\; x_g(z) = g_\beta^r(z)\, d_1^u(z^2). \tag{1.38}$$

But, in turn,

$$e_1^u(z) = e_{1,h}^u(z)+e_{1,g}^u(z) \text{ where } e_{1,h}^u(z) = h^r(z)e_2^u(z^2), \;\; e_{1,g}^u(z) = g_\beta^r(z)\, d_2^u(z^2). \tag{1.39}$$

Substituting (1.39) into (1.38), we get

$$x(z) = x_{hh}(z) + x_{gh}(z) + x_g(z) \text{ where } x_{hh}(z) = h^r(z)h^r(z^2)e_2^u(z^4),$$

$$x_{gh}(z) = h^r(z)g_\beta^r(z^2)\, d_2^u(z^4).$$

Hence, the signal is expanded as follows

$$x(l) = \sum_{k\in\mathbb{Z}} e_2^u(k)\, \varphi^{r,2}(l-4k) + \sum_{k\in\mathbb{Z}} d_2^u(k)\, \psi_\beta^{r,2}(l-4k) + \sum_{k\in\mathbb{Z}} d_1^u(k)\, \psi_\beta^{r,1}(l-2k) \tag{1.40}$$

where low- and high-frequency reconstruction wavelets of the second scale are defined as

$$\varphi^{r,2}(l) = \sum_{k\in\mathbb{Z}} \varphi^{r,1}(k)\varphi^{r,1}(l-2k), \quad \psi_\beta^{r,2}(l) = \sum_{k\in\mathbb{Z}} \psi_\beta^{r,1}(k)\varphi^{r,1}(l-2k).$$

The coordinates in (1.40) are inner products with 4-sample shifts of the decomposition wavelets of the second scale:

$$\tilde\varphi_\beta^{r,2}(l) = \sum_{k\in\mathbb{Z}} \tilde\varphi_\beta^{r,1}(k)\tilde\varphi_\beta^{r,1}(l-2k), \quad \tilde\psi_\beta^{r,2}(l) = \sum_{k\in\mathbb{Z}} \tilde\psi^{r,1}(k)\tilde\varphi_\beta^{r,1}(l-2k).$$

Namely,

$$e_2^u(k) = \langle \mathbf{x}, \tilde\varphi_{\beta,k}^2 \rangle, \quad \text{where } \tilde\varphi_{\beta,k}^{r,2}(l) = \tilde\varphi_\beta^{r,2}(l-4k)$$
$$d_2^u(k) = \langle \mathbf{x}, \tilde\psi_{\beta,k}^{r,2} \rangle, \quad \text{where } \tilde\psi_{\beta,k}^{r,2}(l) = \tilde\psi_\beta^{r,2}(l-4k).$$

7. Experimental results

We conducted a wide series of experiments on compression of various images using a number of wavelet transforms constructed above.

7.1. Employed wavelet transforms

In this section we specify the transforms which we employed in our experiments. The transforms are implemented in a lifting mode as it is described in Section 2. For prediction we used splines of orders 2, 4 and 6. For the lifting step we chose the control filter β as it was explained in Section 3.2. We denote by T_p^r the transform where the decomposition is: follows:

$$d_1^u(z) = d_1(z) - zF_r(z)e_1(z) \tag{1.41}$$

$$e_1^u(z) = e_1(z) + \frac{1}{2}F_p(z)z^{-1}\, d_1^u(z) \tag{1.42}$$

and the reconstruction – in reverse order. Here, $F_q(z^2) = zU_q(z)$.

We conducted experiments using seven types of devised wavelet transforms: T_1^1, T_2^1, T_1^2, T_2^2, T_3^2, T_2^3, T_3^3. In Figure 1.1 we display impulse responses and frequency responses of the low-pass synthesis and analysis filters used by these transforms. We recall that the low-pass synthesis filters coincide with the bi-directional low-pass Butterworth filters. In Figure 1.2 we display impulse responses and frequency responses of the high-pass synthesis and analysis filters. The low-pass analysis filters coincide with the bi-directional High -pass Butterworth filters. We compared between the obtained results of the above filters and the popular B9/7 biorthogonal filters [20].

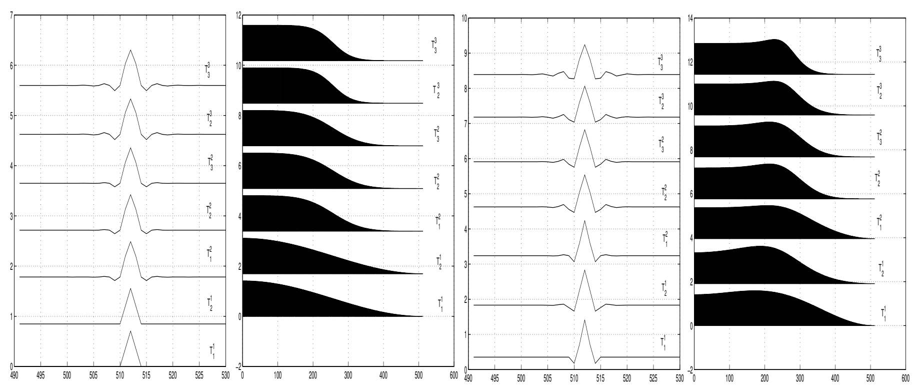

Figure 1.1. The left pair bottom to top: Impulse and frequency responses of the low-frequency synthesis filters h^r used by the transforms T_1^1, T_2^1, T_1^2, T_2^2, T_3^2, T_2^3, T_3^3. The right pair bottom to top: Impulse and frequency responses of the low-frequency analysis filters $\tilde{h}^r_\beta$.

T_1^1 **transform:** This is a single that uses only FIR filters. The transfer functions of the filters are:

$$\tilde{g}^1(z) = -\frac{1-z^2}{2z^2}, \quad \tilde{h}^1\beta(z) = \frac{-z^2+2z+6+2/z-1/z^2}{8}.$$

It is readily seen that this transform coincides with the spline biorthogonal transform of order 2-2 by [20]. Both the analysis and synthesis high-frequency wavelets have 2 vanishing moments. We display in Figure 1.3 the synthesis and analysis wavelets related to the transform T_1^1 and their Fourier spectra.

T_2^1 **transform:** The high-pass analysis filter $\tilde{g}^1$ and the low-pass synthesis filter h^1 are FIR filters. Hence, the low-frequency synthesis

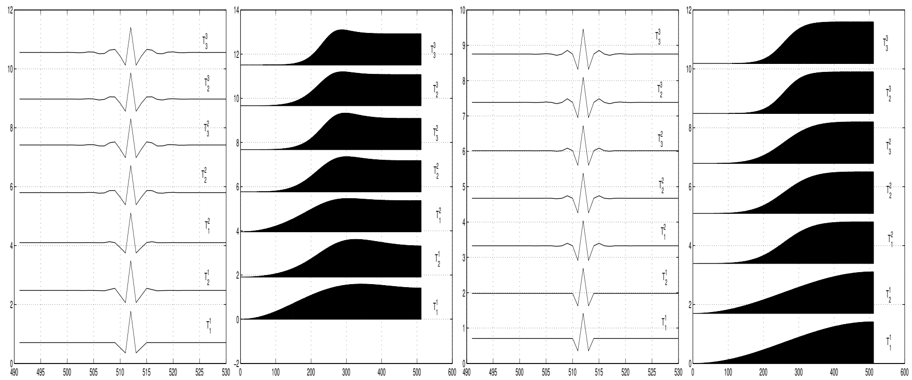

Figure 1.2. The left pair bottom to top: Impulse and frequency responses of the high-frequency synthesis filters g^r_β used by the transforms T^1_1, T^1_2, T^2_1, T^2_2, T^2_3, T^3_2, T^3_3. The right pair bottom to top: Impulse and frequency responses of the high-frequency analysis filters $\tilde{g}^r$.

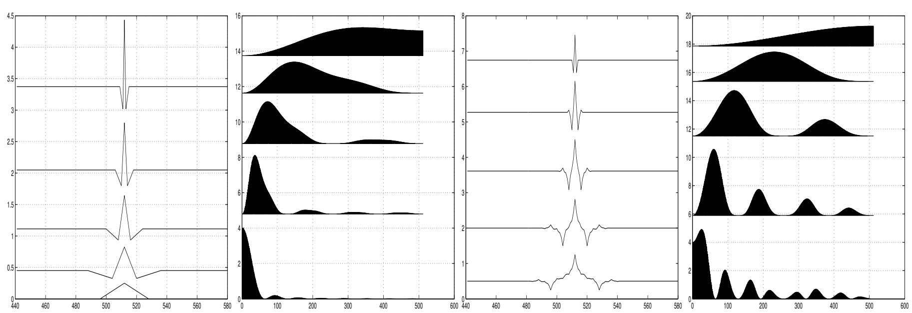

Figure 1.3. The left pair bottom to top: The synthesis wavelets used by the transformT^1_2: $\varphi^{1,4}$ and $\psi^{1,k}_\beta$, $k = 4,3,2,1$ and their Fourier spectra. The right pair bottom to top: The analysis wavelets used by the transformT^1_1: $\tilde{\varphi}^{1,4}_\beta$ and $\psi^{1,k}_\beta$, $k = 4,3,2,1$ and their Fourier spectra.

wavelets $\varphi^{1,s}$, $s = 1,2,\ldots$ and the high-frequency analysis wavelets $\tilde{\psi}^{1,1}$ are compactly supported. We display in Figure 1.4 the synthesis and analysis wavelets related to the transform T^1_1 and their Fourier spectra. The analysis wavelets are obviosly more regular than those in the previos transform. Both the analysis and synthesis high-frequency wavelets have 2 vanishing moments.

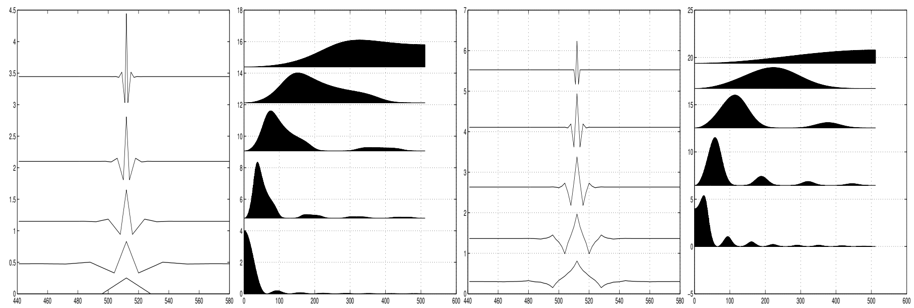

Figure 1.4. The left pair bottom to top: The synthesis wavelets used by the transformT_2^1: $\varphi^{1,4}$ and $\psi_\beta^{1,k}$, $k = 4,3,2,1$ and their Fourier spectra. The right pair bottom to top: The analysis wavelets used by the transformT_2^1: $\tilde{\varphi}_\beta^{1,4}$ and $\psi_\beta^{1,k}$, $k = 4,3,2,1$ and their Fourier spectra.

T_1^2 **transform:** The FIR filters are used in the update step of decomposition and the corresponding step of reconstruction. Tthe other used IIR filters. We display in Figure 1.5 the synthesis and analysis wavelets related to the transform T_1^2 and their Fourier spectra. The analysis high-frequency wavelets have 4 vanishing moments and the synthesis ones – only two.

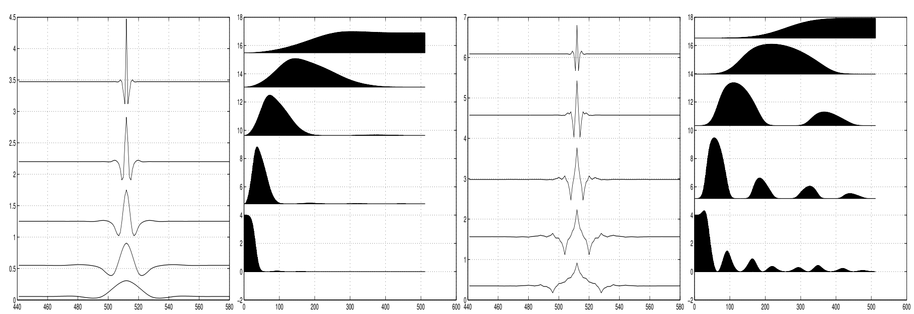

Figure 1.5. The left pair bottom to top: The synthesis wavelets used by the transformT_1^2: $\varphi^{2,4}$ and $\psi_\beta^{2,k}$, $k = 4,3,2,1$ and their Fourier spectra. The right pair bottom to top: The analysis wavelets used by the transformT_1^2: $\tilde{\varphi}_\beta^{2,4}$ and $\psi_\beta^{2,k}$, $k = 4,3,2,1$ and their Fourier spectra.

T_2^2 **transform:** Starting from this transform only IIR filters are used. We display in Figure 1.6 the synthesis and analysis wavelets related to the transform T_2^2 and their Fourier spectra. One can observe that the wavelets are regular and, to some extent, are similar to the synthesis ones. Both the analysis and synthesis high-frequency wavelets have 4 vanishing moments.

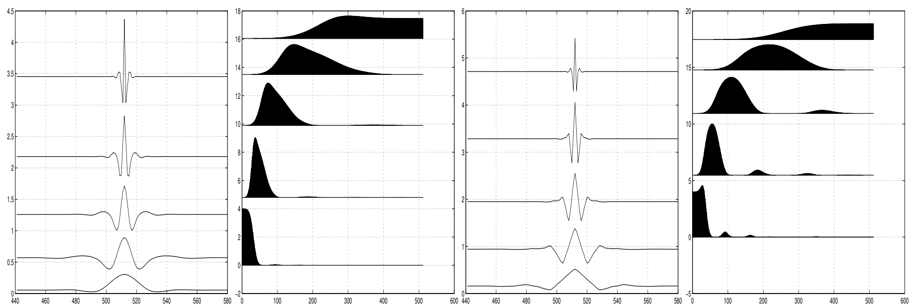

Figure 1.6. The left pair bottom to top: The synthesis wavelets used by the transformT_2^2: $\varphi^{2,4}$ and $\psi_\beta^{2,k}$, $k = 4,3,2,1$ and their Fourier spectra. The right pair bottom to top: The analysis wavelets used by the transformT_2^2: $\tilde{\varphi}_\beta^{2,4}$ and $\psi_\beta^{2,k}$, $k = 4,3,2,1$ and their Fourier spectra.

T_3^2 **transform:** Here the update step of the decomposition uses more complicated filters than the predict step. We display in Figure 1.7 the synthesis and analysis wavelets related to the transform T_3^2 and their Fourier spectra. Both the analysis and synthesis high-frequency wavelets have 4 vanishing moments.

T_2^3 **transform:** Unlike the previous transform, the predict step of the decomposition uses a filter of third order whereas the update step – of the second order. We display in Figure 1.8 the synthesis and analysis wavelets related to the transform T_3^2 and their Fourier spectra. The analysis high-frequency wavelets have 6 vanishing moments, the synthesis one – only 4.

T_3^3 **transform:** As in the transform T_2^2, here the predict step of the decomposition uses the same filter as the update step. We display in Figure 1.9 the synthesis and analysis wavelets related to the transform T_3^2 and their Fourier spectra. Both the analysis and synthesis high-frequency wavelets have 6 vanishing moments and the respective ones are similar to each other. The wavelets are most regular amongst all presented examples. The spectra of the wavelets are well localized in the frequency domain.

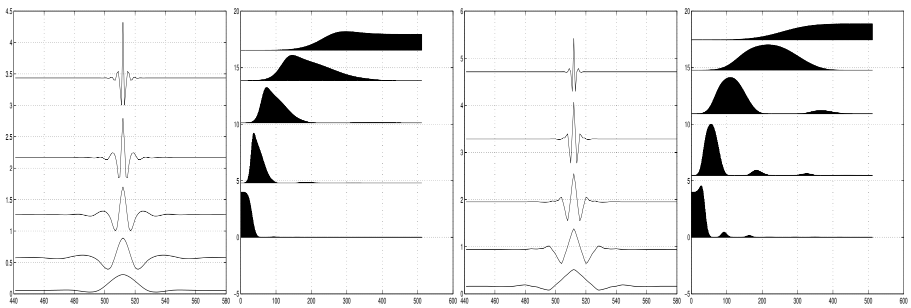

Figure 1.7. The left pair bottom to top: The synthesis wavelets used by the transformT_3^2: $\varphi^{2,4}$ and $\psi_\beta^{2,k}$, $k = 4,3,2,1$ and their Fourier spectra. The right pair bottom to top: The analysis wavelets used by the transformT_3^2: $\tilde{\varphi}_\beta^{2,4}$ and $\psi_\beta^{2,k}$, $k = 4,3,2,1$ and their Fourier spectra.

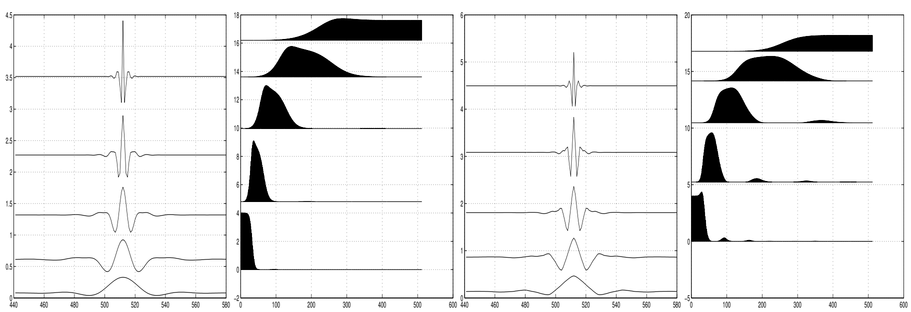

Figure 1.8. The left pair bottom to top: The synthesis wavelets used by the transformT_2^3: $\varphi^{3,4}$ and $\psi_\beta^{3,k}$, $k = 4,3,2,1$ and their Fourier spectra. The right pair bottom to top: The analysis wavelets used by the transformT_2^3: $\tilde{\varphi}_\beta^{3,4}$ and $\psi_\beta^{3,k}$, $k = 4,3,2,1$ and their Fourier spectra.

7.2. Computational complexity of the transforms

The wavelet transforms, which we presented above, are implemented by filtering of even and odd subarrays with filters F_r, $r = 1,2,3$. The filter F_1 has FIR, whereas F_2, F_3 have IIR and rational transfer functions. The latter can be implemented in either cascade or parallel modes. We compare the performance of the transforms with the performance of the wavelet using the B9/7 biorthogonal filter.

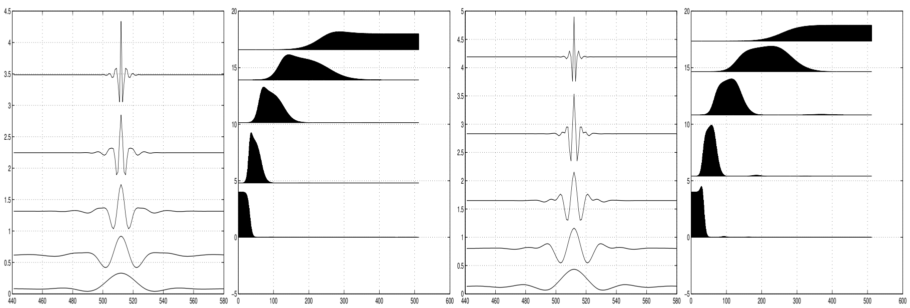

Figure 1.9. The left pair bottom to top: The synthesis wavelets used by the transformT_3^3: $\varphi^{3,4}$ and $\psi_\beta^{3,k}$, $k = 4,3,2,1$ and their Fourier spectra. The right pair bottom to top: The analysis wavelets used by the transformT_3^3: $\tilde{\varphi}_\beta^{3,4}$ and $\psi_\beta^{3,k}$, $k = 4,3,2,1$ and their Fourier spectra.

B9/7 filter:. This filter can be implemented in a fast way using factorization of the filters which was suggested in [4, 22]. Briefly, one step of the decomposition comprises the following operations on even $e_1(k)$ and odd $d_1(k)$ subarrays of the signal $x(k)$ of length N:

$$\begin{aligned}
&d_1(k) = d_1(k) + (e_1(k) + e_1(k+1))\alpha_1,\\
&e_1(k) = e_1(k) + (d_1(k) + d_1(k-1))\alpha_2,\\
&d_1^u(k) = d_1(k) + (e_1(k) + e_1(k+1))\alpha_3,\\
&e_1^u(k) = e_1(k) + (d_1^u(k) + d_1^u(k-1))\alpha_4,\\
&\alpha_1 = -1.586134342,\ \alpha_2 = -0.05298011854,\\
&a_3 = 0.8829110762,\ a_4 = 1.149604398.
\end{aligned}$$

Therefore, one step of 2-D decomposition of an image of size N^2 pixels requires $8A + 4M$ (8 additions plus 4 multiplications) operations per pixel.

T_p^r transforms . These transforms are implemented by filtering followed by:

$$d_1^u(k) = d_1(k) - F^r e_1(k+1),\ e_1^u(k) = e_1(k) + F^p d_1^u(k-1).$$

The number of operations required for filtering of an array of length $N/2$ by the filter $F^1(z) = (1+z)/2$ is $(1A + 1M)N/2$.

If the filter F^2 is implemented in a cascade mode by the formulas (1.32) then the number of operations is $(3A + 3M)N/2$. However, with two processors the filtering can be implemented in a parallel which corresponds to (1.33). Then, each processor performs only $(2A+2M)N/2$ operations.

Implementation of the filter F^3 in a cascade mode using (1.34) requires $(5A+5M)N/2$ operations. With three processors the filtering is implemented in a parallel which corresponds to Eq. (1.35). Then each processor performs only $(3A+2M)N/2$ operations. Moreover, this filtering can be conducted with integers.

Keeping in mind the above considerations we summarize the number of operations per pixel for the employed transforms in Table 1.1. In the first line we give the number of operations for the cascade implementation while and in the second line – the number of operations in each processor in parallel computation.

Tr.	B9/7	T_2^2	T_1^2	T_2^3	T_3^3	T_1^1
Cas.	8A+4M	8A+6M	6A+4M	10A+8M	12A+10M	4A+2M
Par.	–	6A+4M	5A+3M	7A+4M	8A+4M	4A+2M

Table 1.1. Number of operations per pixel for the employed wavelet transforms. First line: cascade implementation. Second line details the number of operations in each processor in the paralllel implementation. The number of operations for T_2^1 is equal to that of T_1^2. The number of operations for T_3^2 is equal to that of T_2^3.

Thus, one can see that even by cascade implementation the cost of the transforms T_2^1, T_1^2 and T_1^1 is lower than the cost of the transform B9/7. Of course, in parallel with 2 or 3 processors we get faster implemenatation than B9/7.

7.3. Experiments with image compression

We conducted a wide series of experiments with compression of multimedia images using the above transforms. In most experiments they outperform the popular B9/7 biorthogonal filter bank. In this section we present the results after compression and decompression of of four images which are displayed in Figures 1.10 and 1.11. These are 512×512 8 bit (8bpp) images. The following experiments were done:

1 We decomposed the image with the wavelet transform up to 6-th scale using the B9/7 filters and the filters listed in Section 7.1.

2 The transform coefficients were coded using the SPIHT algorithm by Said and Pearlman [34]. This algorithm enables to compress data exactly to a prescribed rate. We coded the coefficients with

Figure 1.10. Left: "Lena", right: "Barbara".

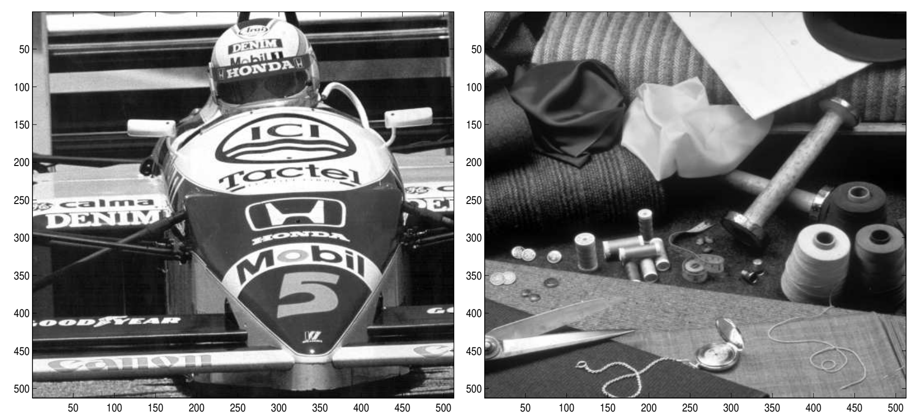

Figure 1.11. Left: "Car", right: "Fabrics".

compression ratios (CR) 1:10 (0.8 bpp), 1:20 (0.4 bpp), 1:30 (4/15 bpp), 1:40 (0.2 bpp) and 1:50 (4/25 bpp).

3 The reconstructed image was compared with the original image and the peak signal to noise ratio (PSNR) in decibels was computed.

$$PSNR = 10 \log_{10} \left(\frac{N\, 255^2}{\sum_{k=1}^{N} (x(k) - \tilde{x}(k))^2} \right) \; dB, \qquad (1.43)$$

Lena: The PSNR values of "Lena" are presented in Table 1.2. One can

CR	B9/7	T_2^2	T_1^2	T_2^1	T_2^3	T_3^3	T_3^2	T_1^1
10	37.70	37.82	37.63	37.41	37.80	37.84	37.85	37.26
20	34.53	34.71	34.54	34.27	34.70	34.72	34.75	34.07
30	32.33	32.62	32.50	32.25	32.62	32.70	32.65	32.03
40	31.42	31.63	31.48	31.30	31.64	31.66	31.67	31.06
50	30.70	30.91	30.74	29.99	30.90	30.92	30.95	29.82

Table 1.2. PSNR of the "Lena" image after the application of SPIHT where the decomposition of the wavelet transform was into 6 scales.

observe that the transform T_2^2 outperforms the B9/7 filter in any compression rate. We recall that the computational complexity of this transform is slightly higher than that of the B9/7 filter but using parallel mode it can be implemented faster. The transforms T_2^3, T_3^3 and T_3^2 outperform the B9/7 filters. On the other hand, they are computationally more expensive than the transform T_2^2. With the parallel mode they can be implemented faster than the B9/7 filter . It is interesting to note that even the transform T_1^2, whose complexity is lower than that of the B9/7, produces better PSNR for almost all the CR. In Figure 1.12 we display the "Lena" image reconstructed from 1:40 compressed files of wavelet coefficients of the B9/7 and T_1^2 transforms. In Figure 1.13 we display the closed-up fragments of these images.

Barbara: The PSNR values of "Barbara"are presented in Table 1.3. From the table we see that the transform T_2^2 outperforms the B9/7

CR	B9/7	T_2^2	T_1^2	T_2^1	T_2^3	T_3^3	T_3^2	T_1^1
10	33.01	33.39	32.92	32.15	33.59	33.72	33.51	31.69
20	28.93	29.13	28.75	28.53	29.24	29.32	29.24	28.18
30	26.99	27.03	26.77	26.71	27.13	27.32	27.15	26.40
40	25.78	25.74	25.30	25.50	25.83	25.94	25.83	25.32
50	25.10	25.00	24.76	24.73	25.09	25.13	25.05	24.57

Table 1.3. PSNR of the "barbara" image after the application of SPIHT where the decomposition of the wavelet transform was into 6 scales.

filters up to 30 compression ratio. For compression ratio of 40 and 50 B9/7 slightly outperforms T_2^2. Unlike T_2^2, the transform T_3^3,

Figure 1.12. Left: "Lena": Reconstructed from 40:1 compression. the B7/9 filter was applied: PSNR=31.42. Right: T_1^2 transform: PSNR=31.47.

Figure 1.13. Closed up fragments of the images from Figure 1.12 .

which can be implemented in integers in fast parallel mode, outperforms the B9/7 transform for all compression ratios. we display in Figure 1.14 the reconstructed "barbara" from the 1:40 compressed files of wavelet coefficients of the B9/7 and T_2^1 transforms.

Car: The PSNR values of "car" are presented in Table 1.4. We can see from the table that the T_2^2 transform outperforms the B9/7 filter for any compression ratio.

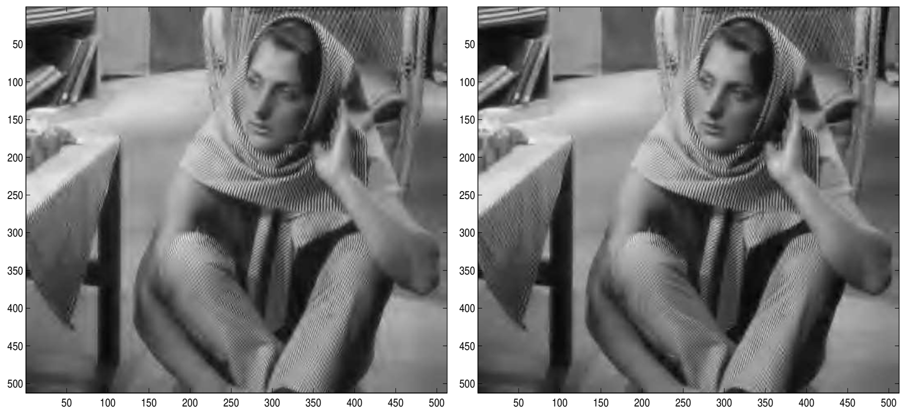

Figure 1.14. Left: "barbara": Reconstructed from 40:1 compression. The B7/9 filter was applied: PSNR=25.78. Right: T_2^1 transform: PSNR=25.50.

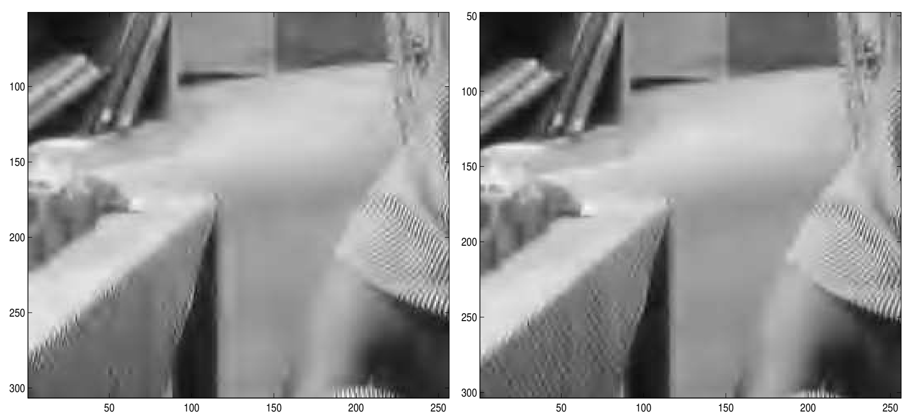

Figure 1.15. Closed up fragments of the image from Figure 1.14 .

Fabric: The PSNR values of "Fabric"are presented in Table 1.5. As it was for the above examples, the transform T_2^2 outperforms the B9/7 filter for any compression ratio. The same is true for more complicated transforms such as T_3^3 and T_3^2. The latter transform produces the best PSNR amongst all listed transforms. Again, PSNR values for the simplified transform T_1^2, are comparable to as those for the B9/7 transform. We display in Figure 1.18 the

CR	B9/7	T_2^2	T_1^2	T_2^1	T_2^3	T_3^3	T_3^2	T_1^1
10	32.57	32.63	32.43	32.61	32.52	32.52	32.66	32.51
20	28.38	28.53	28.36	28.21	28.42	28.46	28.52	28.03
30	26.78	26.99	26.77	26.71	26.88	26.92	27.01	26.47
40	25.05	25.19	25.04	24.95	25.14	25.15	25.21	24.77
50	24.40	24.52	24.34	24.33	24.46	24.46	24.56	24.12

Table 1.4. PSNR of the "car" image after the application of SPIHT where the decomposition of the wavelet transform was into 6 scales.

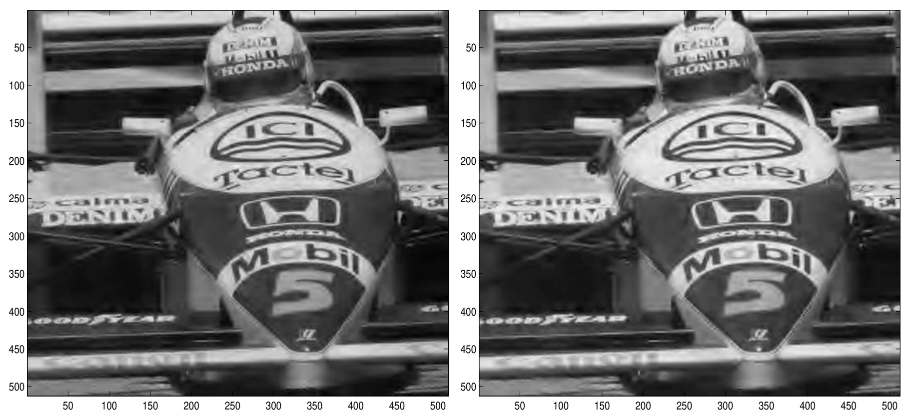

Figure 1.16. Left: "car": Reconstructed from 40:1 compression. the B7/9 filter was applied: PSNR=25.05. Right: T_1^2 transform: PSNR=25.04.

CR	B9/7	T_2^2	T_1^2	T_2^1	T_2^3	T_3^3	T_3^2	T_1^1
10	34.95	34.96	34.79	34.54	34.89	34.97	34.99	34.40
20	31.53	31.53	31.37	31.24	31.46	31.53	31.55	31.09
30	29.62	29.79	29.65	29.68	29.71	29.75	29.81	29.49
40	28.90	29.00	28.86	28.84	28.97	29.02	29.02	28.66
50	28.41	28.52	28.39	28.34	28.34	28.52	28.54	28.15

Table 1.5. PSNR of the "Fabric" image after the application of SPIHT where the decomposition of the wavelet transform was into 6 scales.

image "Fabrics" reconstructed from the 1:40 compressed files of wavelet coefficients of the B9/7 and T_1^2 transforms. The closed up fragments are given in Figure 1.19.

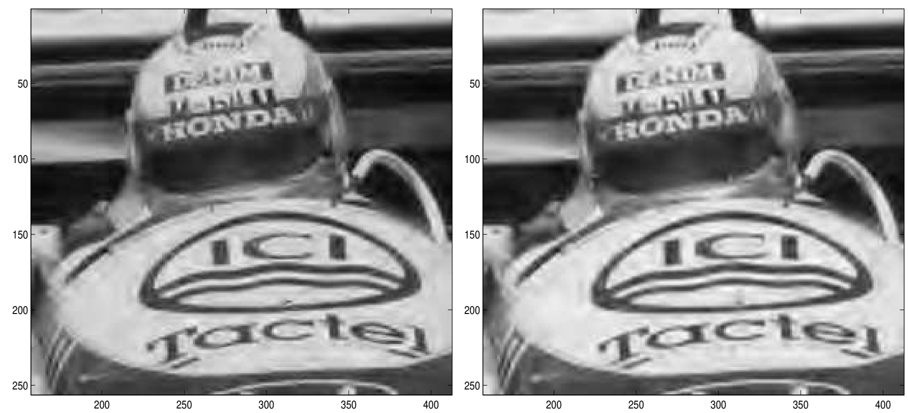

Figure 1.17. Closed up fragments of images from Figure 1.16 .

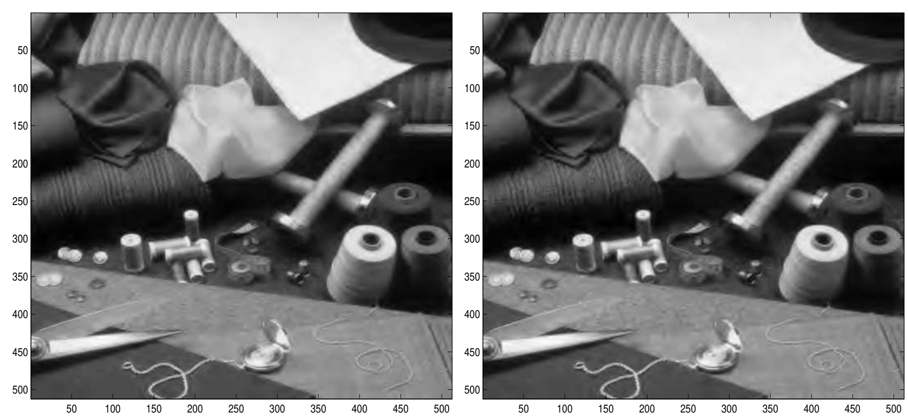

Figure 1.18. Left: "Fabric": Reconstructed from 40:1 compression. the B7/9 filter was applied: PSNR=28.90. Right: T_1^2 transform: PSNR=28.86.

8. Conclusions

In this paper we proposed a new efficient method that produces new filters for the compression of multimedia images. These wavelet transforms were designed by the usage of discrete interpolatory splines. These filters outperform the traditional biorthogonal 9/7 filters which are frequenty used in wavelet based compression. The new filters and the

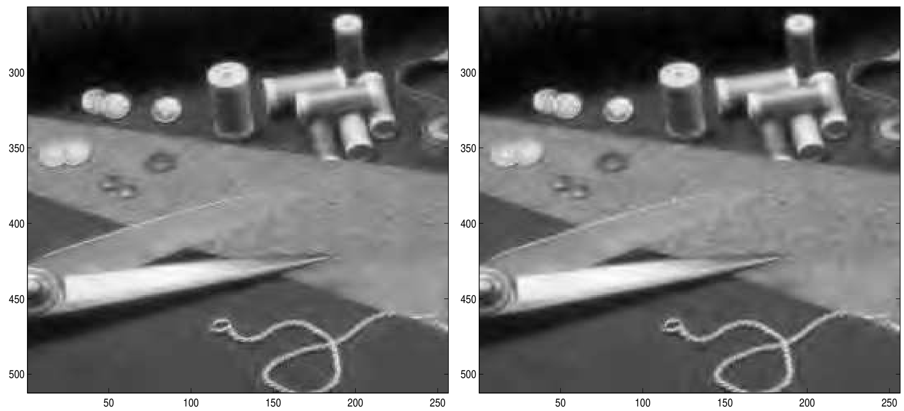

Figure 1.19. Closed up fragments of images from Figure 1.18 .

biorthogonal 9/7 are incorporated into SPIHT in order to measure and compare their performance with one well known codec.

In the future we plan to check the performance of these filters on video and seismic data. In addition, we plan to check their success on other compression methods such as EZW.

References

[1] E. H. Adelson, E. Simoncelli, and R. Hingorani, Orthogonal pyramid transforms for image coding, *Proc. SPIE,* vol. 845, Cambridge, MA, Oct. 1987, pp. 50-58.

[2] M. Antonini, M. Barlaud, P. Mathieu and I. Daubechies, Image Coding using Wavelet Transform, *IEEE Transaction on Image Processing,* 1(2):205-220, 1992.

[3] A. Averbuch, D. Lazar, M. Israeli, Image compression using wavelet transform and multiresolution decomposition, *IEEE Trans. on Image Processing*, 5 : 4:15, Jan. 1996.

[4] A. Z. Averbuch, F. Meyer, J-O Stromberg, Fast Adaptive Wavelet Packet Image Compression, *IEEE Trans. on Image Processing*, 9: 792-800, May 2000.

[5] A. Averbuch, R. Nir, *Still Image Compression using Coded Multiresolution Tree*, Technical Report, Tel Aviv University, 1995.

[6] A. Averbuch, M. Israeli, F. Meyer, Speed vs. Quality in Low Bit-Rate Still Image Compression, it Signal Processing: Image Communication, 15:231-254, 1999.

[7] A. Averbuch, A. Pevnyi, V. Zheludev A lifting scheme of biorthogonal wavelet transform based on discrete interpolatory splines, *Proc. SPIE Wavelet Applications in Signal and Image Processing VIII,* (A. Aldroubi, A. F. Laine; M. A. Unser; Eds.) 4119:564–575, 2000.

[8] A. Z. Averbuch, A. B. Pevnyi and V. A. Zheludev Butterworth wavelets derived from discrete interpolatory splines: Recursive implementation, to appear in *Signal Processing.*, www.math.tau.ac.il/~amir (~zhel).

[9] A. Averbuch, V. Zheludev Construction of biorthogonal discrete wavelet transforms using interpolatory splines, to appear in *Applied and Comp. Harmonic Analysis*, www.math.tau.ac.il/~amir (~zhel).

[10] A. Averbuch, R. Coifman, F. Meyer, *Multi-layered Image Transcription: Application to a Universal Lossless*, submitted.

[11] G. Battle, A block spin construction of ondelettes. Part I. Lemarié functions, *Comm. Math. Phys.* 110:601–615, 1987.

[12] Bing-Bing Chai, Jozsef Vass, and Xinhua Zhuang, Significance-Linked Connected Component Analysis for Wavelet Image Coding, *IEEE Transaction on Image Processing* 8:774-784, June 1999.

[13] M. Boliek, M. J. Gormish , E. L. Schwartz and A. Keith, Next Generation Image Compression and Manipulation Using CREW, *Proc. IEEE ICIP* , 1997, http://www.crc.ricoh.com/CREW http://www.crc.ricoh.com/ gormish/pdf http://www.crc.ricoh.com/CREW/CREW.summary.html

[14] R. Buccigrossi and E. P. Simoncelli, EPWIC: Embedded Predictive Wavelet Image Coder, GRASP Laboratory, TR number 414, http://www.cis.upenn.edu/butch/EPWIC/index.html.

[15] Calderbank, R. C., Daubechies, I., Sweldens, W., and Yeo, B. L. Wavelet Transforms that Map Integers to Integers, Applied and Computational Harmonic Analysis (ACHA), vol. 5, no. 3, pp. 332-369, 1998, http://cm.bell-labs.com/who/wim/papers/integer.pdf.

[16] C. Chrysafis and A. Ortega, Efficient context-based entropy coding for lossy wavelet image compression, *DCC, Data Compression Conference,* Snowbird, UT, March 25 - 27, 1997. (.ps version available from http://biron.usc.edu/ chrysafi/Publications.html)

[17] C. K. Chui and J. Z. Wang, On compactly supported spline wavelets and a duality principle, *Trans. Amer. Math. Soc.* 330:903-915, 1992.

[18] R. L. Claypoole Jr., J. M. Davis, W. Sweldens, R. Baraniuk, Nonlinear wavelet transforms for image coding via lifting, submitted to *IEEE Trans. Image Proc.*.

[19] Coifman, R. R. and Wickerhauser, M. V., Entropy Based Algorithms for Best Basis Selection, *IEEE Trans. Information Theory,* 38:713-718, Mar. 1992.

[20] A. Cohen, I. Daubechies and J.-C. Feauveau, Biorthogonal bases of compactly supported wavelets, *Commun. on Pure and Appl. Math.*' 45:485-560, 1992.

[21] I. Daubechies, *Ten lectures on wavelets*, SIAM. Philadelphia, PA, 1992.

[22] I. Daubechies, W. Sweldens, Factoring wavelet transforms into lifting steps, *J. Fourier Anal. Appl.* 4: 247–269, 1998.

[23] D. L. Donoho, *Interpolating wavelet transform,* Preprint 408, Department of Statistics, Stanford University, 1992.

[24] J. Froment and S. Mallat, Second generation compact image coding with wavelets. in: *Wavelets: A Tutorial in Theory and Applications,* C.K. Chui, editor, vol. 2, Academic Press, NY, 1992.

[25] C. Herley and M. Vetterli, Wavelets and recursive filter banks, *IEEE Trans. Signal Proc.,*' 41(12):2536-2556 1993.

[26] R. Joshi, H. Jafarkhani, J. Kasner, T. Fischer, N. Farvardin, M. Marcellin and R. Bamberger, Comparison of Different Methods of Classification in Subband Coding of Images, *IEEE Trans. Image Proc.,* 6:1473-1487, November 1997.

[27] P. G. Lemarié, Ondelettes à localisation exponentielle, *J. de Math. Pures et Appl.* 67:227-236, 1988.

[28] S. M. LoPresto, K. Ramchandran, and M.T. Orchard, Image coding based on mixture modeling of wavelet coefficients and a fast estimation-quantization framework, it IEEE Data Compression Conference '97 Proceedings, pp. 221-230, March 1997.

[29] D. Marpe and H.L. Cycon, Efficient Pre-Coding Techniques for Wavelet-Based Image Compression, submitted to PCS, Berlin, 1997. For more information, please see http://www.fhtw-berlin.de/Projekte/Wavelet/

[30] D. Marpe, G. Heising, A. P. Petukhov, and H. L. Cycon, Video coding using a bilinear image warping motion model and wavelet-based residual coding, *Proc. SPIE Conf. on Wavelet Applications in Signal and Image Processing*, 3813: 401 – 408, 1999.

[31] A. V. Oppenheim, R. W. Shafer, *Discrete-time signal processing,* Englewood Cliffs, New York, Prentice Hall, 1989.

[32] A. P. Petukhov, Biorthogonal wavelet bases with rational masks and their applications, *Proc. of St. Petersburg Math. Soc.,* Vol. 7:168 – 193, 1999 (Russian).

[33] A. B. Pevnyi and V. A. Zheludev, On the interpolation by discrete splines with equidistant nodes, *J. Appr. Th.,* 102:286-301, 2000.

[34] A. Said and W. W. Pearlman, A new, fast and efficient image codec based on set partitioning in hierarchical trees, *IEEE Trans. on Circ. and Syst. for Video Tech.*, 6: 243-250, June 1996.

[35] S. Servetto, K. Ramchandran, M. Orchard, Image Coding Based on a Morphological Representation of Wavelet Data, *IEEE Transactions on Image Processing,* 8:1161 -1174 Sept. 1999.

[36] J. Shapiro, Embedded image coding using zerotrees of wavelet coefficients, *IEEE Tran. on Signal Processing,* 41: 3445-3462, December 1993.

[37] W. Sweldends The lifting scheme: A custom design construction of biorthogonal wavelets, *Appl. Comput. Harm. Anal.* 3(2):186-200, 1996.

[38] Taubman, D. High Performance Scalable Image Compression with EBCOT, *IEEE Tran. on Image Processing,* 9:1158 -1170, July 2000.

[39] Tsai, M. J., Villasenor, J. D., and Chen, F. Stack-Run Image Coding, *IEEE Trans. CSVT,* 6(5):519-521, Oct. 1996.

[40] M. Unser, A. Aldroubi and M. Eden, A family of polynomial spline wavelet transforms, *Signal Processing*, 30:141-162, 1993.

[41] Saha, S. and Vemuri, R. Adaptive Wavelet Coding of Multimedia Images, Proc. ACM Multimedia Conference, Nov. 1999.

[42] G. Strang, and T. Nguen, *Wavelets and filter banks*, Wellesley-Cambridge Press, 1996.

[43] V. A. Zheludev, Periodic splines and the fast Fourier transform, *Comput. Math. Math. Phys.,* 32(2):149-165, 1992.

[44] Xiong, Z., Ramachandran, K. and Orchard, M. T. *Space-Frequency Quantization for Wavelet Image Coding*, IEEE Trans. IP, vol. 6, no. 5, May 1997, pp. 677-693,

[45] V. A. Zheludev, Wavelet analysis in spaces of slowly growing splines via integral representation, *Real Analysis Exchange,* 24:229-261, 1998/99.

[46] W/IEC JTC1/SC29/WG1 N505, Call for contributions for JPEG2000 (ITC 1.29.14, 15444): Image coding system, 1997.

[47] International Organisation for Standardisation (ISO). Call for contributions for JPEG 2000 (JTC 1.29.14, 15444): Image Coding System, March 1997. ISO/IEC JTCI/SC29/WG1 N505.