

WAILI: Wavelets with Integer Lifting

Geert Uytterhoeven Filip Van Wulpen
Maarten Jansen Dirk Roose Adhemar Bultheel

Report TW 262, July 1997



Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

WAILI: Wavelets with Integer Lifting

Geert Uytterhoeven Filip Van Wulpen
Maarten Jansen Dirk Roose Adhemar Bultheel

Report TW 262, July 1997

Department of Computer Science, K.U.Leuven

Abstract

WAILI is a wavelet transform library, written in C++. It includes some basic image processing operations based on the use of wavelets and forms the backbone of more complex image processing operations.

In this paper, we give some mathematical foundations (wavelets, the Lifting Scheme, integer transforms) behind WAILI, and discuss the design and features of WAILI.

Acknowledgements

This research is supported by the Flemish Information Technology Action Program ('Vlaams Actieprogramma Informatietechnologie'), project number ITA/950244.

Keywords : wavelets, lifting, integer transform, image processing, software library.

AMS(MOS) Classification : 41A30, 68U10, 94A12, 65D10.

WAILI: Wavelets with Integer Lifting*

Geert Uytterhoeven[†] Filip Van Wulpen[‡] Maarten Jansen[§] Dirk Roose
Adhemar Bultheel
Department of Computer Science — Katholieke Universiteit Leuven[¶]

Abstract

WAILI is a wavelet transform library, written in C++. It includes some basic image processing operations based on the use of wavelets and forms the backbone of more complex image processing operations.

In this paper, we give some mathematical foundations (wavelets, the Lifting Scheme, integer transforms) behind WAILI, and discuss the design and features of WAILI.

1 Introduction

The first part of this paper discusses some mathematical foundations behind wavelets. We discuss the Lifting Scheme into more detail. This scheme is an algorithm, originally designed to compute second generation wavelets in an efficient way. Later it has been shown that it can be used to generate also the “classical” (bi)orthogonal wavelets of Cohen-Daubechies-Feauveau (CDF) type with several advantages over the classical computational schemes. More precisely, we give an inventory of the lifting steps for some CDF wavelet transforms of type $(1, x)$, $(2, x)$, and $(4, x)$.

Classical wavelet transforms convert floating point numbers to floating point numbers. However, in many multimedia applications (images, video, audio) the input data consists of integer values only. An advantage of the Lifting Scheme is that it can be converted easily into a transform that maps integers to integers, while retaining the perfect reconstruction property. Some implementation aspects of this transform are also discussed.

The second part discusses the design and features of *WAILI*, a software library that implements the integer wavelet transforms described in the first part. It provides wavelet transforms and wavelet based operations on two-dimensional images. The library is written in C++.

*wavelets@cs.kuleuven.ac.be, <http://www.cs.kuleuven.ac.be/~wavelets/>

[†]Geert.Uytterhoeven@cs.kuleuven.ac.be, <http://www.cs.kuleuven.ac.be/~geert/>

[‡]Filip.VanWulpen@cs.kuleuven.ac.be, <http://www.cs.kuleuven.ac.be/~filip/>

[§]Maarten.Jansen@cs.kuleuven.ac.be, <http://www.cs.kuleuven.ac.be/~maarten/>

[¶]Celestijnenlaan 200A, B-3001 Heverlee, Belgium

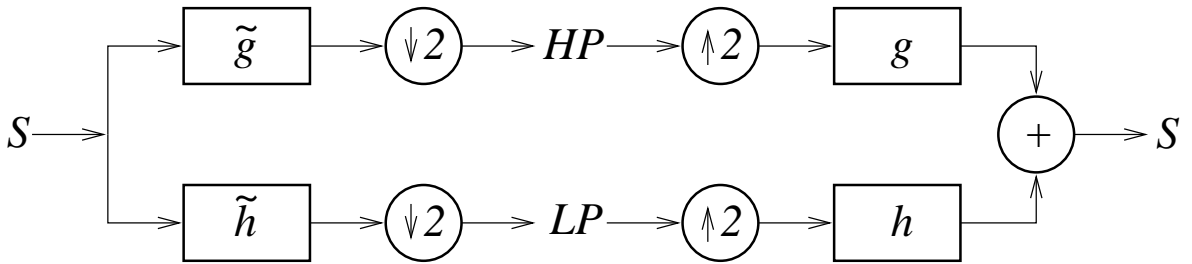


Figure 1: The filter bank algorithm: the signal S is filtered and downsampled to get a low pass signal LP and a high pass signal HP . It can be reconstructed by upsampling and filtering with the correct filters.

2 The Discrete Wavelet Transform

2.1 The Filterbank Algorithm

The basic block in a discrete wavelet transform [10] is a filter bank, consisting of 2 filters (cfr. fig. 1). A discrete signal S is filtered by a high pass filter \tilde{g} and a low pass filter \tilde{h} , and downsampled. The results are a high pass signal HP and a low pass signal LP , each containing half as much samples as the input signal S .

For the inverse transform, first the signals HP and LP are upsampled by putting zeroes in between every sample. After that they are filtered by the filters g and h and the result is added together. In the case of perfect reconstruction — we only consider perfect reconstruction filter banks here — the resulting signal is equal to the original signal S .

The filters g , h , \tilde{g} and \tilde{h} are called *wavelet filters* if they fulfill certain conditions. The filters \tilde{g} and \tilde{h} are called dual filters w.r.t. g and h .

The signal LP resembles the original signal S very much: it is a representation of S at a lower resolution level. On the other hand, HP contains the detail information that is lost by making the transition to that lower resolution level [12].

If the low frequency information is treated as a new signal and passed through the same filter bank, and this step is repeated several times, after some iterations a very low frequency signal is retained. Together with the detail information for the different resolution levels, it represents the original signal decomposed in several resolution levels. This is called a *wavelet transform*.

2.2 Polyphase Representations

The z -transform of a signal $x = \{x_k\}$ is defined as

$$x(z) = \sum_k x_k z^{-k},$$

whenever this makes sense. Thus, the z -transform of a finite impulse response (FIR) filter $h = \{h_{k_b}, \dots, h_{k_e}\}$ is a Laurent polynomial $h(z)$ given by

$$h(z) = \sum_{k=k_b}^{k_e} h_k z^{-k}.$$

Assuming $h_{k_b}, h_{k_e} \neq 0$, we define the degree of such a Laurent polynomial as

$$|h(z)| = k_e - k_b.$$

For consistency we set the degree of the zero polynomial equal to $-\infty$.

Filtering a signal x by a filter h is easily described in the z -domain as an ordinary multiplication: $y(z) = h(z)x(z)$.

Subsampling a signal $x = \{x_k\}$ corresponds to keeping only the even samples $x^s = \{x_{2k}\}$. Thus its z -transform is given by

$$x^s(z) = x_e(z) = \sum_k x_{2k} z^{-k}$$

where $x_e(z^2)$ is the even part of $x(z)$:

$$x_e(z^2) = \frac{x(z) + (x-z)}{2}.$$

Defining the odd part $x_o(z^2)$ in a similar way

$$x_o(z^2) = \frac{z}{2} [x(z) - x(-z)] = \sum_k x_{2k+1} z^{-2k}$$

we can decompose $x(z)$ as

$$x(z) = x_e(z^2) + z^{-1}x_o(z^2).$$

The filtering part of the filterbank decomposition described above can be written as

$$\begin{bmatrix} lp(z) \\ hp(z) \end{bmatrix} = \begin{bmatrix} h(z) \\ g(z) \end{bmatrix} x(z).$$

while the subsampling step corresponds to

$$\begin{aligned} LP(z^2) &= \frac{lp(z) + lp(-z)}{2} = l_{p_e}(z^2), \\ HP(z^2) &= \frac{hp(z) + hp(-z)}{2} = l_{p_o}(z^2). \end{aligned}$$

This can be globalized as

$$\begin{bmatrix} LP(z^2) \\ HP(z^2) \end{bmatrix} = \frac{1}{2} \begin{bmatrix} h(-z) & h(z) \\ g(-z) & g(z) \end{bmatrix} \begin{bmatrix} x(-z) \\ x(z) \end{bmatrix}.$$

However, by first filtering and then subsampling, we first compute all the coefficients and then throw away half of the work done. It would be more efficient if the subsampling step could be done before the filtering, which means that we only compute the even parts of l_p and h_p . Obviously

$$l_{p_e} = [h(z)x(z)]_e = h_e(z)x_e(z) + z^{-1}h_o(z)x_o(z).$$

Thus $\lambda(z) = l_{p_e}(z)$ and $\gamma(z) = h_{p_e}(z)$ are obtained as

$$\begin{bmatrix} \lambda(z) \\ \gamma(z) \end{bmatrix} = P(z) \begin{bmatrix} x_e(z) \\ z^{-1}x_o(z) \end{bmatrix},$$

where $P(z)$ is the *polyphase matrix*

$$P(z) = \begin{bmatrix} h_e(z) & h_o(z) \\ g_e(z) & g_o(z) \end{bmatrix}.$$

The *lazy wavelet transform* is the most simple wavelet transform. It does nothing but splitting the signal in even and odd components. Its polyphase matrix is the unit matrix:

$$P_{\text{azy}}(z) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

3 The Lifting Scheme

The Lifting Scheme [15, 13, 14] found its roots in a method to improve a given wavelet transform to obtain some specific properties.

3.1 Primal and Dual Lifting

With *primal lifting*, starting from two complementary finite filters h and g , a new finite filter h^{new} complementary to g is created:

$$h^{\text{new}}(z) = h(z) + s(z^2)g(z),$$

or, using the polyphase representation:

$$P^{\text{new}}(z) = \begin{bmatrix} h_e^{\text{new}}(z) & h_o^{\text{new}}(z) \\ g_e(z) & g_o(z) \end{bmatrix} = \begin{bmatrix} 1 & s(z) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} h_e(z) & h_o(z) \\ g_e(z) & g_o(z) \end{bmatrix}.$$

With *dual lifting*, starting from two complementary finite filters h and g , a new finite filter g^{new} complementary to h is created:

$$g^{\text{new}}(z) = g(z) + t(z^2)h(z),$$

or, using the polyphase representation:

$$P^{\text{new}}(z) = \begin{bmatrix} h_e(z) & h_o(z) \\ g_e^{\text{new}}(z) & g_o^{\text{new}}(z) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ t(z) & 1 \end{bmatrix} \begin{bmatrix} h_e(z) & h_o(z) \\ g_e(z) & g_o(z) \end{bmatrix}$$

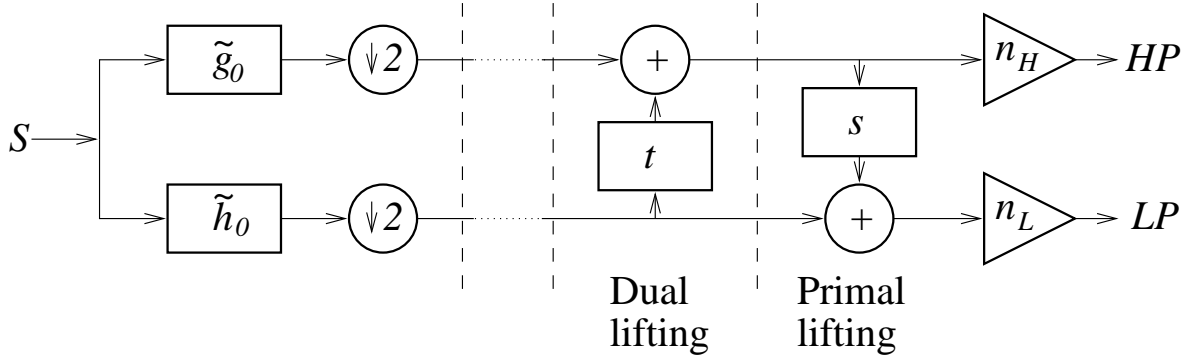


Figure 2: The Lifting Scheme: starting from wavelet filters \tilde{g}_0 and \tilde{h}_0 we *lift* the signal to obtain another wavelet decomposition. The final step is a normalization with factors n_H and n_L .

This process can be repeated, obtaining a *cakewalk* of alternating primal and dual lifting steps, as shown in figure 2. If the initial $P(z)$ is the unit matrix — i.e. we start from the lazy wavelet transform — the result is a product of unit upper and lower triangular 2×2 matrices of Laurent polynomials.

3.2 Decomposition into Lifting Steps

Daubechies and Sweldens [4] proved that *any* polyphase matrix representing a wavelet transform with finite filters can be factored in a product of unit upper and lower triangular 2×2 matrices, and a diagonal normalization matrix:

$$P(z) = \underbrace{\begin{bmatrix} K_1 & 0 \\ 0 & K_2 \end{bmatrix}}_{\text{normalization}} \prod_{i=m}^1 \left\{ \overbrace{\begin{bmatrix} 1 & s_i(z) \\ 0 & 1 \end{bmatrix}}^{\text{primal lifting}} \underbrace{\begin{bmatrix} 1 & 0 \\ t_i(z) & 1 \end{bmatrix}}_{\text{dual lifting}} \right\}.$$

with $s_i(z)$ and $t_i(z)$ Laurent polynomials and K_1 and K_2 normalization factors. It can be shown [4] that, based on an Euclidian type algorithm, it is possible to obtain any wavelet transform by performing a lazy wavelet transform followed by alternating primal and dual lifting steps, and a normalization.

From this it immediately follows that the inverse transform can be written as

$$P^{-1}(z) = \prod_{i=1}^m \left\{ \begin{bmatrix} 1 & 0 \\ -t_i(z) & 1 \end{bmatrix} \begin{bmatrix} 1 & -s_i(z) \\ 0 & 1 \end{bmatrix} \right\} \begin{bmatrix} 1/K_1 & 0 \\ 0 & 1/K_2 \end{bmatrix}.$$

How can we find the $s_i(z)$ and $t_i(z)$? To decompose the polyphase matrix using primal

lifting as

$$P(z) = \begin{bmatrix} h_e(z) & h_o(z) \\ g_e(z) & g_o(z) \end{bmatrix} = \begin{bmatrix} h_e(z) & h_o^{\text{new}}(z) \\ g_e(z) & g_o^{\text{new}}(z) \end{bmatrix} \begin{bmatrix} 1 & s(z) \\ 0 & 1 \end{bmatrix}$$

we have to find Laurent polynomials $s(z)$, $h_o^{\text{new}}(z)$, and $g_o^{\text{new}}(z)$ such that

$$\begin{aligned} h_o(z) &= s(z)h_e(z) + h_o^{\text{new}}(z), \\ g_o(z) &= s(z)g_e(z) + g_o^{\text{new}}(z), \end{aligned}$$

with (recall $|\cdot|$ means degree)

$$\begin{aligned} |h_o^{\text{new}}(z)| &< |h_o(z)|, \\ |g_o^{\text{new}}(z)| &< |g_o(z)|. \end{aligned}$$

This corresponds to two *long division with remainder* schemes of Laurent polynomials, with a common quotient $s(z)$. This corresponds to an Euclidian algorithm applied to $h_o(z)$ and $h_e(z)$. The conditions imposed by $g_o(z)$ and $g_e(z)$ come practically for free as is illustrated in the example below. This is a consequence of the perfect reconstruction condition. For more details see [4]. Note that the division of Laurent polynomials is not unique, so we may have multiple possibilities. The scheme for dual lifting is similar.

Example

The analyzing filterpair for the Cohen-Daubechies-Feauveau biorthogonal wavelets [3] with 2 vanishing moments for both the primal and dual wavelet function is

$$\begin{aligned} \tilde{h}(z) &= -\frac{1}{8}z^{-2} + \frac{1}{4}z^{-1} + \frac{3}{4} + \frac{1}{4}z - \frac{1}{8}z^2, \\ \tilde{g}(z) &= \frac{1}{4}z^{-2} - \frac{1}{2}z^{-1} + \frac{1}{4}. \end{aligned}$$

The corresponding polyphase matrix is

$$P(z) = \begin{bmatrix} -\frac{1}{8}z^{-1} + \frac{3}{4} - \frac{1}{8}z & \frac{1}{4} + \frac{1}{4}z \\ \frac{1}{4}z^{-1} + \frac{1}{4} & -\frac{1}{2} \end{bmatrix}.$$

To decompose this matrix using dual lifting as

$$P(z) = \begin{bmatrix} h_e^{\text{new}}(z) & \frac{1}{4} + \frac{1}{4}z \\ g_e^{\text{new}}(z) & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ t(z) & 1 \end{bmatrix}$$

we have to find Laurent polynomials $t(z)$, $h_e^{\text{new}}(z)$, and $g_e^{\text{new}}(z)$ such that

$$\begin{aligned} -\frac{1}{8}z^{-1} + \frac{3}{4} - \frac{1}{8}z &= h_e^{\text{new}}(z) + t(z) \left(\frac{1}{4} + \frac{1}{4}z \right), \\ \frac{1}{4}z^{-1} + \frac{1}{4} &= g_e^{\text{new}}(z) + t(z) \left(-\frac{1}{2} \right). \end{aligned}$$

Thus we calculate the quotient $t(z)$ and the remainder $h_e^{\text{new}}(z)$ of a long division. This division is not unique: there are three solutions. We have chosen:

$$\begin{array}{r}
 -\frac{1}{8}z^{-1} + \frac{3}{4} - \frac{1}{8}z \quad \left| \begin{array}{l} \frac{1}{4} + \frac{1}{4}z \\ \hline -\frac{1}{2}z^{-1} - \frac{1}{2} \end{array} \right. \\
 -\frac{1}{8}z^{-1} - \frac{1}{8} \\
 \hline
 \phantom{-\frac{1}{8}z^{-1}} + \frac{7}{8} - \frac{1}{8}z \\
 \phantom{-\frac{1}{8}z^{-1}} - \frac{1}{8} - \frac{1}{8}z \\
 \hline
 +1
 \end{array}$$

This choice will allow us to perform further lifting steps later. A solution to this problem is thus given by

$$\begin{aligned}
 t(z) &= -\frac{1}{2}z^{-1} - \frac{1}{2}, \\
 h_e^{\text{new}}(z) &= 1, \\
 g_e^{\text{new}}(z) &= 0,
 \end{aligned}$$

and thus

$$P(z) = \begin{bmatrix} 1 & \frac{1}{4} + \frac{1}{4}z \\ 0 & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{2}z^{-1} - \frac{1}{2} & 1 \end{bmatrix}.$$

We can continue the decomposition to obtain

$$P(z) = \begin{bmatrix} 1 & 0 \\ 0 & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} 1 & \frac{1}{4} + \frac{1}{4}z \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{2}z^{-1} - \frac{1}{2} & 1 \end{bmatrix}.$$

Summarized, the analyzing step can be performed by a dual lifting step, followed by a primal lifting step and a normalization. If we omit the normalization factors K_1 and K_2 , it can be implemented using the following algorithm:

Forward transform		Inverse transform	
Splitting	$s_i \leftarrow x_{2i}$ $d_i \leftarrow x_{2i+1}$	Inverse primal lifting	$s_i \leftarrow s_i - \frac{1}{4}(d_{i-1} + d_i)$
Dual lifting	$d_i \leftarrow d_i - \frac{1}{2}(s_i + s_{i+1})$	Inverse dual lifting	$d_i \leftarrow d_i + \frac{1}{2}(s_i + s_{i+1})$
Primal lifting	$s_i \leftarrow s_i + \frac{1}{4}(d_{i-1} + d_i)$	Merging	$x_{2i} \leftarrow s_i$ $x_{2i+1} \leftarrow d_i$

n	\tilde{n}	
1		$\tilde{g}(z) = \frac{1}{2}z^{-1} - \frac{1}{2}$
		$t_1(z) = -1$
		$K_1 = 1$
	1	$\tilde{h}(z) = \frac{1}{2}z^{-1} + \frac{1}{2}$
		$s_1(z) = \frac{1}{2}$
		$K_2 = \frac{1}{2}$
	3	$\tilde{h}(z) = -\frac{1}{16}z^{-3} + \frac{1}{16}z^{-2} + \frac{1}{2}z^{-1} + \frac{1}{2} + \frac{1}{16}z - \frac{1}{16}z^2$
		$s_1(z) = -\frac{1}{16}z^{-1} + \frac{1}{2} + \frac{1}{16}z$
		$K_2 = \frac{1}{2}$
	5	$\tilde{h}(z) = \frac{3}{256}z^{-5} - \frac{3}{256}z^{-4} - \frac{11}{128}z^{-3} + \frac{11}{128}z^{-2} + \frac{1}{2}z^{-1} + \frac{1}{2} + \frac{11}{128}z - \frac{11}{128}z^2 - \frac{3}{256}z^3$ $+ \frac{3}{256}z^4$
		$s_1(z) = \frac{3}{256}z^{-2} - \frac{11}{128}z^{-1} + \frac{1}{2} + \frac{11}{128}z - \frac{3}{256}z^2$ $K_2 = \frac{1}{2}$

Table 1: Decomposition for the analyzing filter pair of the Cohen-Daubechies-Feauveau biorthogonal wavelets with n (\tilde{n}) vanishing moments for the primal (dual) wavelet.

Other examples

The decompositions in lifting steps of some other wavelet transforms of this family are shown in tables 1, 2 and 3.

4 The Integer Wavelet Transform

4.1 Introduction

In many applications (e.g. image compression and processing) the input data consist of integer samples. In addition the storage and encoding of integer numbers is easier, compared to floating point numbers. Unfortunately all of the above transforms assume that the input samples are floating point values. They return floating point values as wavelet coefficients,

n	\tilde{n}	
2		$\tilde{g}(z) = \frac{1}{4}z^{-2} - \frac{1}{2}z^{-1} + \frac{1}{4}$
		$t_1(z) = -\frac{1}{2}z^{-1} - \frac{1}{2}$
		$K_1 = 1$
2		$\tilde{h}(z) = -\frac{1}{8}z^{-2} + \frac{1}{4}z^{-1} + \frac{3}{4} + \frac{1}{4}z - \frac{1}{8}z^2$
		$s_1(z) = \frac{1}{4} + \frac{1}{4}z$
		$K_2 = -\frac{1}{2}$
4		$\tilde{h}(z) = \frac{3}{128}z^{-4} - \frac{3}{64}z^{-3} - \frac{1}{8}z^{-2} + \frac{19}{64}z^{-1} + \frac{45}{64} + \frac{19}{64}z - \frac{1}{8}z^2 - \frac{3}{64}z^3 + \frac{3}{128}z^4$
		$s_1(z) = -\frac{3}{64}z^{-1} + \frac{19}{64} + \frac{19}{64}z - \frac{3}{64}z^2$
		$K_2 = -\frac{1}{2}$
6		$\tilde{h}(z) = -\frac{5}{1024}z^{-6} + \frac{5}{512}z^{-5} + \frac{17}{512}z^{-4} - \frac{39}{512}z^{-3} - \frac{123}{1024}z^{-2} + \frac{81}{256}z^{-1} + \frac{175}{256}$ $+ \frac{81}{256}z - \frac{123}{1024}z^2 - \frac{39}{512}z^3 + \frac{17}{512}z^4 + \frac{5}{512}z^5 - \frac{5}{1024}z^6$
		$s_1(z) = \frac{5}{512}z^{-2} - \frac{39}{512}z^{-1} + \frac{81}{256} + \frac{81}{256}z - \frac{39}{512}z^2 + \frac{5}{512}z^3$
		$K_2 = -\frac{1}{2}$

Table 2: Decomposition for the analyzing filter pair of the Cohen-Daubechies-Feauveau biorthogonal wavelets with n (\tilde{n}) vanishing moments for the primal (dual) wavelet (cont'd).

n	\tilde{n}	
4		$\tilde{g}(z) = -\frac{1}{16}z^{-3} + \frac{1}{4}z^{-2} - \frac{3}{8}z^{-1} + \frac{1}{4} - \frac{1}{16}z$
		$t_1(z) = 0$
		$s_1(z) = -\frac{1}{4} - \frac{1}{4}z$
		$t_2(z) = -z^{-1} - 1$ $K_1 = 2$
2		$\tilde{h}(z) = \frac{3}{32}z^{-3} - \frac{3}{8}z^{-2} + \frac{5}{32}z^{-1} + \frac{5}{4} + \frac{5}{32}z - \frac{3}{8}z^2 + \frac{3}{32}z^3$
		$s_2(z) = \frac{3}{16} + \frac{3}{16}z$
		$K_2 = -\frac{1}{4}$
4		$\tilde{h}(z) = -\frac{5}{256}z^{-5} + \frac{5}{64}z^{-4} - \frac{1}{256}z^{-3} - \frac{3}{8}z^{-2} + \frac{35}{128}z^{-1} + \frac{35}{32} + \frac{35}{128}z - \frac{3}{8}z^2 - \frac{1}{256}z^3$ $+ \frac{5}{64}z^4 - \frac{5}{256}z^5$
		$s_2(z) = -\frac{5}{128}z^{-1} + \frac{29}{128} + \frac{29}{128}z - \frac{5}{128}z^2$
		$K_2 = -\frac{1}{4}$
6		$\tilde{h}(z) = \frac{35}{8192}z^{-7} - \frac{35}{2048}z^{-6} - \frac{55}{8192}z^{-5} + \frac{115}{1024}z^{-4} - \frac{557}{8192}z^{-3} - \frac{733}{2048}z^{-2} + \frac{2625}{8192}z^{-1}$ $+ \frac{525}{512} + \frac{2625}{8192}z - \frac{733}{2048}z^2 - \frac{557}{8192}z^3 + \frac{115}{1024}z^4 - \frac{55}{8192}z^5 - \frac{35}{2048}z^6 + \frac{35}{8192}z^7$
		$s_2(z) = \frac{35}{4096}z^{-2} - \frac{265}{4096}z^{-1} + \frac{499}{2048} + \frac{499}{2048}z - \frac{265}{4096}z^2 + \frac{35}{4096}z^3$ $K_2 = -\frac{1}{4}$

Table 3: Decomposition for the analyzing filter pair of the Cohen-Daubechies-Feauveau biorthogonal wavelets with n (\tilde{n}) vanishing moments for the primal (dual) wavelet (cont'd).

even if the input values actually were integer. Rounding the floating point values to integer values does not help because then we will lose the perfect reconstruction feature.

Fortunately the lifting scheme can be easily modified to a transform that maps integers to integers and that is reversible, and thus allows a perfect reconstruction [2]. This will be done by adding some rounding operations, at the expense of introducing a non-linearity in the transform.

4.2 Integer Lifting

A lifting step basically looks like

$$x^{\text{new}}(z) \leftarrow x(z) + s(z)y(z).$$

Because the signal part $y(z)$ is not changed by the lifting step, the result of the filter operation can be rounded (indicated by $\{\}$):

$$x^{\text{new}}(z) \leftarrow x(z) + \{s(z)y(z)\},$$

while retaining a 100% reversible operation:

$$x(z) \leftarrow x(z)^{\text{new}} - \{s(z)y(z)\}.$$

This is the most amazing feature of integer lifting: whatever rounding operation is used, the lifting operation will *always* be reversible.

4.3 Rational Lifting Coefficients

In some cases, all filter coefficients of $s(z)$ are rational and the corresponding lifting step can be written as

$$x_i \leftarrow x_i - \frac{1}{a} \sum_j b_j y_j,$$

where $a, b_j \in \mathbf{Z}$, i.e. they are integers. This is true for various wavelet transforms, among which the Cohen-Daubechies-Feauveau biorthogonal wavelets.

Such a lifting step can be modified in one of the following ways:

Full rounding The result of the division by a is rounded:

$$\tilde{x}_i \leftarrow x_i - \left\{ \frac{\sum_j b_j y_j}{a} \right\}.$$

Here \tilde{x}_i is an integer approximation to the x_i obtained with floating point lifting.

Without rounding We avoid the division by a by multiplying the other terms with a :

$$\tilde{x}_i \leftarrow ax_i - \sum_j b_j y_j.$$

Here $\tilde{x}_i = ax_i$, and thus the dynamic range of the wavelet coefficients will increase. This has to be taken into account in later steps. Note that in this case no real rounding is performed, and thus this can be considered to be an “exact” implementation of the floating point version, yielding integers.

Mixed form We combine the rounding and multiplication steps of both methods:

$$\tilde{x}_i \leftarrow a_1 x_i - \left\{ \frac{\sum_j b_j y_j}{a_2} \right\},$$

with

$$\begin{aligned} a_1, a_2 &\in \mathbf{Z}, \\ a_1 \cdot a_2 &= a. \end{aligned}$$

This variant can be used if we want to have more control over the dynamic range of the resulting \tilde{x}_i .

Note that in all of the three cases above the modified lifting step is still reversible, and thus the perfect reconstruction feature is still present.

If $a, b_j \notin \mathbf{Z}$, we can still use the “full rounding” method to obtain a transform that maps integers to integers.

In most cases we will use “full rounding”, except when we want to control the dynamic range of the result.

Examples

CDF (1, x) This is the well-known Haar transform. The integer version looks like:

Forward transform		Inverse transform	
Splitting:	$s_i \leftarrow x_{2i}$ $d_i \leftarrow x_{2i+1}$	Inverse primal lifting:	$s_i \leftarrow s_i - \left\{ \frac{d_i}{2} \right\}$
Dual lifting:	$d_i \leftarrow d_i - s_i$	Inverse dual lifting:	$d_i \leftarrow d_i + s_i$
Primal lifting:	$s_i \leftarrow s_i + \left\{ \frac{d_i}{2} \right\}$	Merging:	$x_{2i} \leftarrow s_i$ $x_{2i+1} \leftarrow d_i$

A graphical representation of the forward transform is shown in figure 3.

CDF (2, x) Our algorithm becomes:

Forward transform		Inverse transform	
Splitting:	$s_i \leftarrow x_{2i}$ $d_i \leftarrow x_{2i+1}$	Inverse primal lifting:	$s_i \leftarrow s_i - \left\{ \frac{1}{4}(d_{i-1} + d_i) \right\}$
Dual lifting:	$d_i \leftarrow d_i - \left\{ \frac{1}{2}(s_i + s_{i+1}) \right\}$	Inverse dual lifting:	$d_i \leftarrow d_i + \left\{ \frac{1}{2}(s_i + s_{i+1}) \right\}$
Primal lifting:	$s_i \leftarrow s_i + \left\{ \frac{1}{4}(d_{i-1} + d_i) \right\}$	Merging:	$x_{2i} \leftarrow s_i$ $x_{2i+1} \leftarrow d_i$

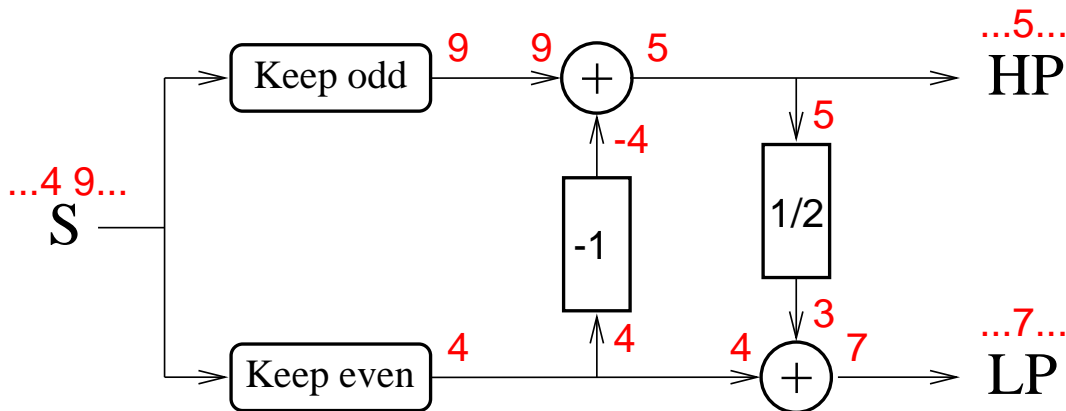


Figure 3: Graphical representation of the integer Haar wavelet transform.

4.4 Normalization

In the above examples, we again omitted the normalization factors K_1 and K_2 . However, in many applications it is important to know the real values of the normalized wavelet coefficients. They indicate the importance of the corresponding wavelet components.

Furthermore, if we want the energy of the signal to be retained in the wavelet coefficients — or at least an approximation in the nonorthogonal case —, we have to use normalized filter coefficients that are $\sqrt{2}$ times as large as the filter coefficients we used. Thus the real normalization factors are $\sqrt{2}K_1$ for the low pass band, and $\sqrt{2}K_2$ for the high pass band. Note that the normalization factors for the low pass band accumulate, due to the iteration on the low pass band.

In other cases, we may want that the mean value of the low pass band coefficients is the same as the mean value of the original signal. This will be true if we make sure that $K_1 = 1$. E.g. if a digital image is transformed, the low pass band will be a low resolution version of the original image. Note that the dynamic range of the pixel values may increase due to the non convexity of the low pass filter.

5 Wavelet Denoising

5.1 Thresholding

An important application domain of wavelet theory is de-noising. A classical wavelet based noise reduction algorithm has the following scheme: it starts with a discrete wavelet transform. Then, the input wavelet coefficients are manipulated and finally an inverse transform yields the result. The *manipulation* of a coefficient often depends on the *class* to which the coefficient belongs. Therefore the algorithm first classifies the coefficients before the actual manipulation. For the allocation of each coefficient to a class, the algorithm uses a *criterion*, which is often binary: it divides the coefficients into two classes. The first group contains important, regular coefficients, while the other group of coefficients is catalogued as “not essential and too much affected by noise”.

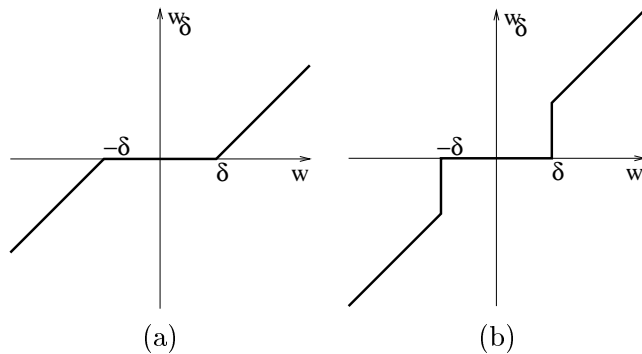


Figure 4: Noise reduction by wavelet shrinking. (a) Soft-thresholding: a wavelet coefficient w with an absolute value below the threshold δ is replaced by 0. Coefficients with higher absolute values are shrunk. (b) Hard-thresholding: Coefficients with an absolute value above the threshold are kept.

The most straightforward procedure uses the *absolute value* of the coefficients as a regularity measure: a coefficient close to zero contains little information and is relatively strongly influenced by noise. The basic idea is the same as for compression algorithms: the method assumes that a regular image can be represented by a small number of large coefficients. Donoho e.a. [7] showed that this method has statistical optimality properties. The absolute value is however a local measure: every coefficient is considered separately. More sophisticated algorithms often use a more global criterion.

We use such a threshold procedure. All wavelet coefficients with absolute value below a certain threshold δ , are classified as “noisy”. The algorithm replaces them by zero. The coefficients above the threshold are shrunk with the same value δ . Figure 4 compares this “soft-thresholding” operation with the “hard-thresholding” alternative. The hard-thresholding procedure does not shrink the coefficients with absolute value above the threshold. Although at first sight this may seem a more natural approach, soft-thresholding is a more continuous operation, and it is mathematically more tractable.

5.2 Threshold selection

The central issue in a threshold procedure is the selection of an appropriate threshold. If this threshold is too small, the result is still noisy. On the other hand, a large threshold also removes important image features and thus causes a bias. It is intuitively clear that the more noise is expected for a given coefficient, the higher the optimal threshold should be. Most procedures compute one threshold for a group of coefficients, based on the statistics of this group. Therefore, they assume additive, stationary noise within this group. If the input noise is additive and stationary, then it is easy to prove that the noise on the wavelet coefficients is additive and stationary within each subband and at each resolution level [11, 8]. This property does not hold exactly true for integer transforms, due to the lack of linearity.

The optimal threshold for subband j $\delta_{j,\text{opt}}$ minimizes the mean square error of the result

as compared with the unknown, noise-free coefficients:

$$R_j(\delta) = \frac{1}{N_j} \|\mathbf{w}_{j,\delta} - \mathbf{v}_j\|^2,$$

where N_j is the number of coefficients in subband j , $\mathbf{w}_{j,\delta}$ is the vector of thresholded noisy coefficients and \mathbf{v}_j is the vector of noise-free coefficients. The expected value of this error is called *Risk* function.

The “universal threshold” δ_j^U by Donoho and Johnstone [5] explicitly proposes a threshold value proportional to the amount of noise σ_j .

$$\delta_j^U = \sqrt{2 \log(N_j)} \sigma_j.$$

In fact, for finite N_j this is an upper bound for the optimal threshold. This choice is asymptotically optimal.

Another well known threshold selection procedure is based on “Stein’s Unbiased Risk Estimator” (SURE) [6].

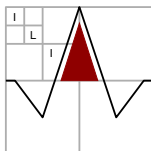
We use the minimizer of the “Generalized Cross Validation” (GCV) function [16, 9, 8]

$$GCV_j(\delta) = \frac{\frac{1}{N_j} \|\mathbf{w}_j - \mathbf{w}_{j,\delta}\|^2}{\left[\frac{N_{j0}}{N_j} \right]^2},$$

where N_{j0} is the number of wavelet coefficients that were replaced by zero. Unlike most other selection procedures it does not need an estimate of the actual noise level σ_j in subband j . Moreover, computational complexity is linear. The method is asymptotically optimal, but for finite N_j it gives better estimates of the optimal threshold than δ_j^U .

Figure 5 illustrates the principle of generalized cross validation for a typical case. Artificial, stationary, colored noise was added to a clean test image. Both the noisy and noise-free image were transformed, using an integer version of the biorthogonal Cohen-Daubechies-Feauveau (2,2)-wavelet algorithm. At each level and for each subband, the GCV as a function of the threshold was compared with the exact mean square error. Although the conditions for a successful application of GCV are not strictly fulfilled by the nonlinear character of the integer transform, both functions have approximately the same minimum. Note that in practical problems, a comparison with the exact mean square error is impossible, since the noise-free image is unknown.

6 WAILI: Wavelets with Integer Lifting



WAILI is a software library — written in C++ — providing wavelet transforms and wavelet based operations on two-dimensional images of various kinds. Applications are situated in image processing.

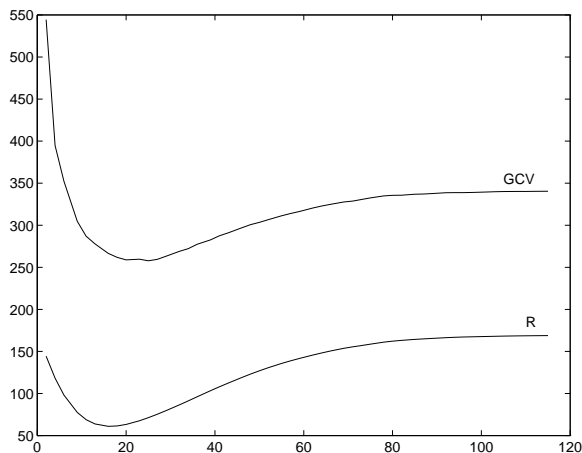


Figure 5: *GCV* and mean square error of the result in function of the threshold δ . Both functions have approximately the same minimum.

6.1 The Lifting Scheme

We chose to implement two-dimensional tensor product wavelet transforms using the integer version of the Lifting Scheme.

Although the Lifting Scheme allows to transform signals with a finite length without extending the signal, we did not choose to take this approach. Instead we use the classical symmetric extension [1] because it is easier to implement and suffices for the applications we have in mind.

All arithmetic operations are done with 16 bit inputs and 16 bit results. This should suffice for applications where the input data is 8 bit wide. Of course this can easily be changed if necessary.

6.2 Cohen-Daubechies-Feauveau Biorthogonal Wavelets

The wavelets we use are a subclass of the Cohen-Daubechies-Feauveau family of biorthogonal wavelets [3]. Their key benefits are:

- They have finite support. This preserves the locality of image features.
- The scaling function $\varphi(x)$ is always symmetric, and the wavelet function $\psi(x)$ is always symmetric or antisymmetric. This is important for image processing operations since the detection of features in an image does not depend on their left/right or up/down orientation.
- Its filter coefficients are dyadic, i.e. they are of the form $z/2^n$, with $z \in \mathbf{Z}$ and $n \in \mathbf{N}$. This simplifies the implementation. But unfortunately this feature is not always preserved by the decomposition in lifting steps.

We chose not to use wavelets with more than 6 vanishing moments to restrict the filter lengths. Longer filters have less locality and thus perform worse in image processing applications, in spite of their increase in smoothness.

WAILI provides the following wavelet transforms of this family ((n, \tilde{n}) means that the primal wavelet has n vanishing moments, while the dual wavelet has \tilde{n} vanishing moments):

(1, x): (1, 1), (1, 3), (1, 5)

(2, x): (2, 2), (2, 4), (2, 6)

(4, x): (4, 2), (4, 4), (4, 6)

We deliberately did not implement any of the $(3, x)$ or $(5, x)$ wavelet transforms because their lifting steps require divisions by 3 or 5 — which are difficult to implement in hardware —, or because the normalization factor K_1 requires a division. $(6, x)$ are not implemented either because they require more than 16 bits (for 8 bit input data).

6.3 Wavelet Transforms

6.3.1 Image Objects

An *Image* consists of one or more independent channels, thus allowing for different sizes and wavelet transform types per channel. No interpretation or format is imposed on the channels and its data. The actual meaning of the image data can be freely chosen by the user. Examples are grayscale, RGB, YUV or Lab color, etc. . . .

6.3.2 Channel Objects

The basic building block of the library is the *Channel*. A channel is a rectangular matrix containing one-valued pixels. A channel can be non-transformed (an *NTChannel*), or wavelet-transformed (an *LChannel* — lifted channel).

Since a wavelet transform is some kind of *recursive* transform, a *LChannel* contains some subchannels (subbands), which can be either non-transformed or wavelet-transformed. The number of subchannels in an *LChannel* depends on the type of wavelet transform. One can have the following combinations:

LChannelCR Obtained by transforming both the columns and rows of an *NTChannel*. As a result, one has 4 subbands:

LL Low pass band in both the horizontal and the vertical direction,

LH Low pass band in the vertical direction, high pass in the horizontal direction,

HL High pass band in the vertical direction, low pass in the horizontal direction,

HH High pass band in both the horizontal and the vertical direction.

LChannelC Obtained by transforming only the columns of an *NTChannel*. As a result, one has 2 subbands:

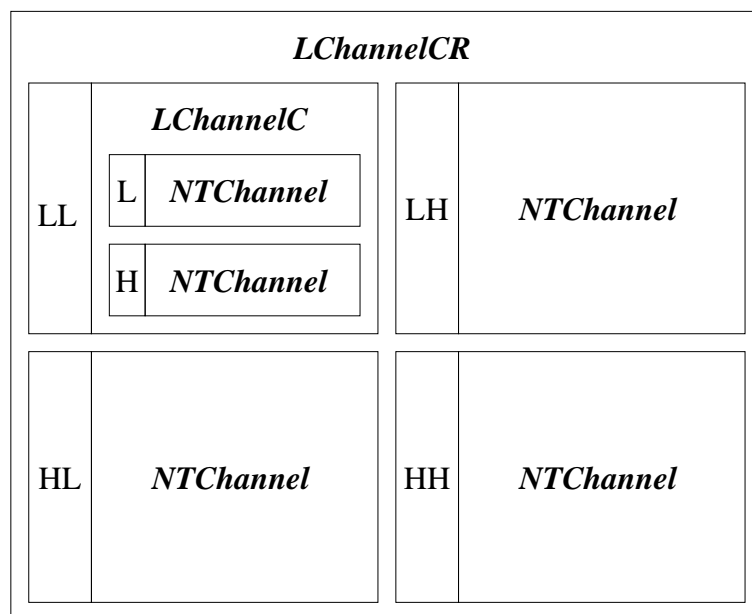


Figure 6: Example of a channel after two transform levels. In the first step both the columns and the rows are transformed, in the second step only the columns are transformed.

L Low pass band in the vertical direction,

H High pass band in the vertical direction.

LChannelR Obtained by transforming only the rows of an *NTChannel*. As a result, one has 2 subbands:

L Low pass band in the horizontal direction,

H High pass band in the horizontal direction.

Fig. 6 shows an example of a channel after two transform levels. And of course one can use a different wavelet at each transform level.

6.3.3 *Wavelet Objects*

A *Wavelet* represents the filters and lifting steps associated with a specific wavelet transform. Some wavelet transforms of the Cohen-Daubechies-Feauveau family are implemented. One can add his own favorite wavelet transform if he has a decomposition in integer lifting steps for it.

7 Wavelet based Operations

7.1 Crop and Merge on Wavelet Transformed Images

The wavelet and scaling functions are local in the spacial and the frequency domain. If one wants to cut a small rectangle B out of a large image A , one can exploit the spacial locality of the basis functions.

At each transform level l , we only need the coefficients that influence the pixel values of B . This is a rectangle of coefficients that is only slightly larger than B (scaled to the transform level l). The extra coefficients are needed due to the overlap of the basis functions at the borders of B .

Using these technique we efficiently implemented the following operations:

Crop Get a rectangular part of an image, at a specified position and with a specified size.

Merge Paste a rectangular image into a larger image, at a specified position.

7.2 The Wavelet Transform and Translation Invariance

Due to the subsampling and the different treatment of even and odd samples in the Lifting Scheme a wavelet transform is not translation invariant: if a signal is delayed or advanced its wavelet transform is not simply a delayed or advanced version of the wavelet transform of the original signal. Only if the delay or advance is a multiple of 2^n , with n the number of transform levels, the wavelet transform will be a delayed or advanced version of the original transformed signal. In two dimensions, it is even worse because this property needs to be true in both the horizontal and vertical directions.

The redundant wavelet transform is translation invariant, but it needs much more memory and processing time, so this is not an option in many applications.

Since we wanted to allow crop and merge operations on wavelet transformed images we came up with the following scheme.

If each transform level is considered independently, one step of a wavelet transform is translation invariant if the translation is limited to an even number of pixels. Thus we associate with every matrix *coordinates* (a horizontal and vertical offset for the upper left pixel) which depend on the transform level. At each transform level we have two versions of the wavelet transform: an *even* and an *odd* version. Which transform is used depends only on the parity of the offset. If the parities of the coordinates match at each level, we can merge two images without retransforming one of them. If they do not match, we have to retransform one of them. The same is true for the crop operation.

The main idea behind this scheme is that in many cases the coordinates of the subimage that will be pasted into another image are *known in advance*, so it can be transformed correctly. An example of this is the creation of one large image by concatenating several separately created subimages.

7.3 Scaling of Images

If a digital image is transformed, the low pass band will be a low resolution version of the original image. This way we can downscale images by a power of 2. Alternatively, we can use the inverse transform — with all high pass coefficients being zero — to upscale an image by a power of 2.

If we want to scale an image with a factor that is not a power of two, we scale the image using a power of 2 factor such that the result is larger than the wanted result. After that a classical interpolation algorithm is used to downscale the image to the wanted size.

7.4 Noise Reduction

Wavelet soft-thresholding is used to reduce additive, stationary noise. We select a threshold for each transform level and for each subband separately. The technique that we use to select these thresholds is generalized cross validation (See section 5.2). Since this is an asymptotical method, it only performs well for sufficiently large subbands. Experiments pointed out that 1000 coefficients in a subband is about the minimum for a good threshold estimation. Therefore, WAILI does not perform the de-noising operation on subbands with less than 1000 coefficients. These are the subbands at coarse levels. In most cases, most of the noise appears at finer levels, and so, leaving the coarse levels untouched generally gives good results.

8 Acknowledgements

This research is supported by the Flemish Information Technology Action Program (‘Vlaams Actieprogramma Informatietechnologie’), project number ITA/950244.

The third author is financed by a grant from the Flemish Institute for the Promotion of Scientific and Technological Research in the Industry (IWT).

References

- [1] C. M. Brislawn. Classification of nonexpansive symmetric extension transforms for multirate filter banks. *Appl. Comput. Harmon. Anal.*, 3:337–357, 1996.
- [2] R. Calderbank, I. Daubechies, W. Sweldens, and B.-L. Yeo. Wavelet transforms that map integers to integers. Technical report, Department of Mathematics, Princeton University, 1996.
- [3] A. Cohen, I. Daubechies, and J. Feauveau. Bi-orthogonal bases of compactly supported wavelets. *Comm. Pure Appl. Math.*, 45:485–560, 1992.
- [4] I. Daubechies and W. Sweldens. Factoring wavelet transforms into lifting steps. Technical report, Bell Laboratories, Lucent Technologies, 1996.
- [5] D. L. Donoho and I. M. Johnstone. Ideal spatial adaptation via wavelet shrinkage. *Biometrika*, 81:425–455, 1994.

- [6] D. L. Donoho and I. M. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *J. Amer. Statist. Assoc.*, 90:1200–1224, 1995.
- [7] D. L. Donoho, I. M. Johnstone, G. Kerkycharian, and D. Picard. Wavelet shrinkage: Asymptopia? *J. Roy. Statist. Soc. Ser. B*, 57(2):301–369, 1995.
- [8] M. Jansen and A. Bultheel. Multiple wavelet threshold estimation by generalized cross validation for images with correlated noise. Submitted for publication, February 1997.
- [9] M. Jansen, M. Malfait, and A. Bultheel. Generalized cross validation for wavelet thresholding. *Signal Processing*, 56(1):33–44, January 1997.
- [10] B. Jawerth and W. Sweldens. An overview of wavelet based multiresolution analyses. *SIAM Rev.*, 36(3):377–412, 1994.
- [11] I. M. Johnstone and B. W. Silverman. Wavelet threshold estimators for data with correlated noise. *J. Roy. Statist. Soc. Ser. B*, 59:319–351, 1997.
- [12] S. G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. Patt. Anal. Mach. Intell.*, 11(7):674–693, 1989.
- [13] W. Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM J. Math. Anal.*, (to appear).
- [14] W. Sweldens. The lifting scheme: A new philosophy in biorthogonal wavelet constructions. In A. F. Laine and M. Unser, editors, *Wavelet Applications in Signal and Image Processing III*, pages 68–79. Proc. SPIE 2569, 1995.
- [15] W. Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Appl. Comput. Harmon. Anal.*, 3(2):186–200, 1996.
- [16] N. Weyrich and G. T. Warhola. De-noising using wavelets and cross validation. In S.P. Singh, editor, *Approximation Theory, Wavelets and Applications*, volume 454 of *NATO ASI Series C*, pages 523–532, 1995.