A global method for invertible integer DCT and integer wavelet algorithms

Gerlind Plonka
Institute of Mathematics
Gerhard-Mercator-University of Duisburg
D – 47048 Duisburg
Germany
E–*mail*: plonka@math.uni-duisburg.de

Dedicated to Manfred Tasche on the occasion of his 60-th birthday

Abstract

In this paper we present a new, simple, global method to derive invertible integer-to-integer mappings from given linear mappings $\hat{F} : \mathbb{R}^n \mapsto \mathbb{R}^n$. If $\hat{F}$ is given by $\hat{F}(\mathbf{x}) = H_n \mathbf{x}$ and H_n is an invertible matrix, then one can always find a suitable factor α such that the condition $[-\frac{1}{2}, \frac{1}{2})^n \subseteq \alpha H_n((\frac{1}{2}, \frac{1}{2}]^n)$ is satisfied. An invertible mapping $F : \mathbb{Z}^n \mapsto \mathbb{Z}^n$ can now simply be defined by $F(\mathbf{x}) = \mathrm{rd}\,(H_n \mathbf{x})$, and obviously, this nonlinear integer mapping is close to the linear mapping $\alpha \hat{F}$. We apply this idea in order to derive a new invertible integer DCT-II transform of radix-2 length and new integer wavelet algorithms. It turns out, that the expansion factors α can be chosen very small.

Mathematics Subject Classification 2000. 65T50, 65G50, 15A23, 94A08.

Key words. Invertible integer to integer transforms, Discrete cosine transform, lossless coding, expansion factors, rounding–off, reversible integer DCT, integer wavelet transform.

1 Introduction

The discrete cosine transform of type II (DCT-II) as well as biorthogonal wavelet transforms have found a wide range of applications in signal processing, especially in image compression. But for lossless coding, all of the transform coefficient bits must be coded to ensure perfect reconstruction of the original signal. Thus, if the input data consist of integer vectors or integer matrices, it is highly desirable to be able to characterize the output again with integers. In other words, one is interested in invertible transforms, which map integers to integers and well approximate their linear counterparts.
Such transforms are called *integer transforms*, or more precisely, integer-to integer transforms. These integer transforms are not fundamentally integer in nature. They can be based on floating point arithmetic or fixed point arithmetic (involving only integer additions, integer multiplications and binary shifts) in conjunction with rounding operations.

Integer wavelet transforms are usually based on the lifting realization of a linear wavelet transform (see e.g. [1, 2, 4, 6, 12, 15]) and rounding off after each lifting step. Earlier approaches like the S-transform [25] and the $S + P$ transform [27] can be seen as special cases of the lifting scheme approach. Alternative methods for constructing reversible integer wavelet transforms are based on the overlapping rounding transform [14] or use ladder networks with IIR filters [17].
Most biorthogonal wavelet filter banks use (after appropriate scaling) only filter coefficients which are dyadic rationals. These filter banks are especially suitable for fixed-point arithmetic. However, also filters with irrational coefficients (like Daubechies orthogonal filters) may lead to integer transforms, either using floating point arithmetic in the lifting steps and rounding off (see [4]) or using fixed point arithmetic after rounding of filter coefficients to dyadic rationals.
The performance of integer wavelet transforms has been compared to their conventional counterparts for lossy compression in a number of experiments (see [1, 2]). But up to now, no exact comparisons with the linear wavelet transform (in terms of error estimates for the coefficients) are known. Clearly, since each lifting step further causes the approximation error to increase, transforms with fewer lifting steps tend to perform better. An important source of error is caused by rounding the intermediate results to integers.

Integer DCT algorithms are usually based on a factorization of the (scaled) cosine matrix C_n^{II} into a product of simple matrices (containing only integers or dyadic rationals) and so-called lifting matrices, whose diagonal elements are 1, and only one nondiagonal element in nonzero. Then, applying the lifting technique (similarly as in the wavelet case) and rounding off, the reversible integer DCT is derived (see e.g. [7, 16, 23]). Analogously, also integer DFT transforms can be found [20].
For fixed-point arithmetic DCT transforms, the real entries in the lifting matrices are replaced by suitable dyadic rationals, thereby reducing the arithmetical complexity of the corresponding algorithms (see e.g. [5, 7, 8, 16, 18, 21, 31]). A statistical approach to find optimal approximants for the real entries in the lifting matrices using the mean square error minimization is proposed in [7, 18]. Observe, that in different papers on integer DCT, the lifting technique is used without roundoff, resulting in a vector of dyadic rationals instead of integers (see e.g. [8, 18, 21, 31]). These are in fact no integer-to integer transforms that we have in mind.

For integer transforms still working in floating point arithmetic, the arithmetical complexity is comparable with those of their underlying linear transforms (see e.g. [23]).
For detailed analysis of the error, when the exact DCT-II transform is compared with integer DCT algorithms (based on floating point arithmetic), we refer to [23]. Replacing the floating point arithmetic by fixed point arithmetic (rounding the cosine values appropriately), one can obtain similarly small worst case errors, if the range for the input vectors is limited (see [24]).

In [13], a general approach for deriving reversible integer mappings from given invertible transforms is presented. The idea can be described as follows. First, the transformation matrix is factorized into triangular matrices, where only integers occur in the diagonals. Then a generalized lifting method and rounding off (N-point reversible transform, see also [16]) is applied in order to obtain the integer transform.

In this paper we want to present a new global approach to derive integer transforms from given linear transforms. The main idea can be seen as a generalization of the expansion factor method in [23] (see also [4]). For a linear transform $\hat{F} : \mathbb{R}^n \to \mathbb{R}^n$ given by $\hat{F}(\mathbf{x}) = H_n\mathbf{x}$, where $H_n \in \mathbb{R}^{n\times n}$ is an invertible matrix, one can find an invertible integer transform $F : \mathbb{Z}^n \to \mathbb{Z}^n$ approximating $\hat{F}$ by

$$F(\mathbf{x}) = \mathrm{rd}\,(H_n\mathbf{x}),$$

if H_n satisfies the *expansion condition* $[-\frac{1}{2}, \frac{1}{2})^n \subseteq H_n\,((-\frac{1}{2}, \frac{1}{2}]^n)$. But usually, the transform matrix H_n does not satisfy this condition. In this case one can blow up the matrix H_n with a suitable expansion factor $\alpha_n > 1$ such that $\alpha_n\, H_n((-\frac{1}{2}, \frac{1}{2}]^n)$ completely covers the unit cube $[-\frac{1}{2}, \frac{1}{2})^n$. Then, an invertible integer transform is simply given by

$$F(\mathbf{x}) = \mathrm{rd}\,(\alpha_n H_n\mathbf{x}).$$

This nonlinear integer transform is clearly very close to the exact (scaled) transform $\alpha_n\, H_n\mathbf{x}$.
In Section 2, we describe the approach for arbitrary invertible linear mappings.
In Section 3, we apply the idea in order to derive a new integer DCT-II algorithm. It appears, that we just need to apply a usual fast DCT-II algorithm to the input data to compute $\alpha_n\, C_n^{II}\mathbf{x}$, where α_n is a relatively small constant depending on the DCT length n. Afterwards, we just round each component of the resulting vector to the next integer.
In Section 4, we apply the approach to the periodic biorthogonal wavelet transform through L levels. Again, we obtain a simple integer wavelet transform by using the fast wavelet transform with a scaling factor α_L and rounding off. The invertibility is kept if α_L is chosen suitably. It turns out the α_L only depends on the number of levels L but not on the length of the input vector n. The method is described for various examples of biorthogonal filter banks.

The proposed integer transform has two advantages. Firstly, the fast algorithms being already implemented for the DCT-II and for wavelet transforms can be directly applied. Secondly, the difference between the integer transform and the exact (scaled) linear transform is exactly controlled.
Observe, that the lifting method can be applied as before in order to obtain a fast algorithm for the wavelet transform, but the rounding off of intermediate results is dropped. Further, one can work in fixed-point arithmetic (by appropriate scaling and/or approximating of coefficients by dyadic rationals. Still, the resulting integer transform will be very close to the corresponding linear mapping.

2 Integer transforms from linear transforms

Let $n \geq 2$ be a given integer. Let $H_n \in \mathbb{R}^{n\times n}$ be an invertible matrix and let $\hat{F} : \mathbb{R}^n \to \mathbb{R}^n$ be the corresponding linear mapping

$$\hat{F}(\mathbf{x}) := H_n\,\mathbf{x}, \qquad \mathbf{x} := (x_j)_{j=0}^{n-1} \in \mathbb{R}^n.$$

If the input vector $\mathbf{x}$ is in $\mathbb{Z}^n$, we look for a nonlinear, invertible transform $F : \mathbb{Z}^n \to \mathbb{Z}^n$, such that $\hat{F}$ is well approximated by F. More precisely, we want to find F in a way that the error

$$\mathbf{e}(\mathbf{x}) := \hat{F}(\mathbf{x}) - F(\mathbf{x})$$

is controlled and remains to be small in each component.
The global method, which we want to propose to solve this problem, can be seen as an extension of the integer transform with expansion factors given in [23].
We use the following notations. For $a \in \mathbb{R}$ let

$$\lfloor a \rfloor := \max\{k \leq a;\, k \in \mathbb{Z}\}, \quad \mathrm{rd}\,(a) := \lfloor a + 1/2 \rfloor, \quad \{a\} := a - \mathrm{rd}\,(a) \in [-\tfrac{1}{2}, \tfrac{1}{2}).$$

We apply these operations to vectors $\mathbf{a} = (a_0, \ldots, a_{n-1})^T \in \mathbb{R}^n$ componentwisely, i.e. $\mathrm{rd}\,(\mathbf{a}) := (\mathrm{rd}\,(a_0), \ldots, \mathrm{rd}\,(a_{n-1}))^T \in \mathbb{Z}^n$ and so on.
Further, let $H_n((-\frac{1}{2},\, \frac{1}{2}]^n) := \{H_n\,\mathbf{r} :\, \mathbf{r} \in (-\frac{1}{2},\, \frac{1}{2}]^n\}$ the image of $(-\frac{1}{2},\, \frac{1}{2}]^n$ under the linear mapping $\hat{F}$ generated by H_n.

Theorem 2.1 *Let $H_n \in \mathbb{R}^{n\times n}$ be an invertible matrix satisfying the* expansion condition

$$[-\tfrac{1}{2},\, \tfrac{1}{2})^n \subseteq \bigcup_{\mathbf{k}\in\ {}^n} (H_n((-\tfrac{1}{2},\, \tfrac{1}{2}]^n) + \mathbf{k}) \tag{2.1}$$

and let $\hat{F}$ be given by $\hat{F}(\mathbf{x}) = H_n\mathbf{x}$. Then, a nonlinear transform $F : \mathbb{Z}^n \to \mathbb{Z}^n$ approximating $\hat{F}$ can be found as follows:
For arbitrary fixed $\mathbf{x} \in \mathbb{Z}^n$ choose a vector $\mathbf{k}_\mathbf{x} \in \mathbb{Z}^n$ such that $\{H_n\mathbf{x}\} \in H_n((-\frac{1}{2},\, \frac{1}{2}]^n) + \mathbf{k}_\mathbf{x}$, and define

$$F(\mathbf{x}) := \mathrm{rd}\,(H_n\mathbf{x}) + \mathbf{k}_\mathbf{x}.$$

Then the nonlinear mapping F is invertible, and we obtain

$$\mathbf{x} = \mathrm{rd}\,(H_n^{-1}(F(\mathbf{x})).$$

Moreover, the error estimates

$$\|\hat{F}(\mathbf{x}) - F(\mathbf{x})\|_2 \leq \left(\sum_{j=0}^{n-1} (\tfrac{1}{2} + |(k_\mathbf{x})_j|^2)\right)^{1/2}, \quad \|\hat{F}(\mathbf{x}) - F(\mathbf{x})\|_\infty \leq \max_{0\leq j\leq n-1} (\tfrac{1}{2} + |(k_\mathbf{x})_j|)$$

hold, where $\mathbf{k}_\mathbf{x} = ((k_\mathbf{x})_j)_{j=0}^{n-1}$.

Proof. The expansion condition (2.1) ensures, that for arbitrary $\mathbf{x} \in \mathbb{Z}^n$ there exists (at least) one vector $\mathbf{k}_\mathbf{x} \in \mathbb{Z}^n$ and a vector $\mathbf{r} \in (-\frac{1}{2},\, \frac{1}{2}]^n$ with

$$\{H_n\mathbf{x}\} = H_n\mathbf{r} + \mathbf{k}_\mathbf{x}.$$

Now, putting $F(\mathbf{x}) := \mathrm{rd}\,(H_n\mathbf{x}) + \mathbf{k_x}$, we find

$$F(\mathbf{x}) = \mathrm{rd}\,(H_n\mathbf{x}) + \mathbf{k_x} = \mathrm{rd}\,(H_n\mathbf{x}) + \{H_n\mathbf{x}\} - H_n\mathbf{r} = H_n(\mathbf{x}-\mathbf{r}).$$

Hence,

$$\mathrm{rd}\,(H_n^{-1}F(\mathbf{x})) = \mathrm{rd}\,(H_n^{-1}\,H_n(\mathbf{x}-\mathbf{r})) = \mathrm{rd}\,(\mathbf{x}-\mathbf{r}) = \mathbf{x}$$

since $\mathbf{r} \in (-\frac{1}{2}, \frac{1}{2}]^n$ and $\mathbf{x} \in \mathbb{Z}^n$, and the mapping F is invertible. The error estimates directly follow from the definition of F. $\square$

The vector $\mathbf{k_x}$ in Theorem 2.1 may be not uniquely determined. Considering the error estimates, we are naturally interested in vectors $\mathbf{k_x}$ with small norm.

If an invertible matrix does not satisfy the expansion condition (2.1) then one can apply an expansion factor $\alpha > 0$ such that $\alpha\, H_n$ (instead of H_n) satisfies the expansion condition. Since $H_n((-\frac{1}{2}, \frac{1}{2}]^n)$ is an n-dimensional parallelepiped with volume $|\det H_n|$, a suitable expansion factor α can always be found.

We are especially interested in trigonometric transforms and in wavelet transforms, where the transform matrices are orthogonal or they are invertible and have a small matrix norm. Especially, orthogonal matrices do *not* satisfy the expansion condition (2.1) and need to be multiplied with a suitable expansion factor. Further, in order to find a simple nonlinear mapping, we would like to have the same vector $\mathbf{k} = \mathbf{k_x}$ for all $\mathbf{x}$. Since $H_n((-\frac{1}{2}, \frac{1}{2}))$ is an symmetric area around zero, $\mathbf{k}$ should be the zero vector.

Applying Theorem 2.1, we can restate our problem as follows: For a given matrix (trigonometric matrix or wavelet matrix) H_n generating the linear mapping $\hat{F} : \mathbf{x} \mapsto H_n\mathbf{x}$, we want to find a minimal expansion factor $\alpha > 0$, such that

$$[-\tfrac{1}{2}, \tfrac{1}{2})^n \subseteq \alpha H_n((-\tfrac{1}{2}, \tfrac{1}{2}]^n).$$

For a given matrix $A = (a_{i,j})_{i,j=0}^{n-1} \in \mathbb{R}^{n\times n}$ let

$$\|A\|_\infty := \max_{0\le i\le n-1} \sum_{j=0}^{n-1} |a_{i,j}|$$

be the maximum row sum norm of $\mathbf{A}$. Then we have

Theorem 2.2 *Let $H_n \in \mathbb{R}^{n\times n}$ be an invertible matrix and $\hat{F} : \mathbb{R}^n \to \mathbb{R}^n$ with $\hat{F}(\mathbf{x}) := H_n\mathbf{x}$ the linear mapping generated by H_n. Then the nonlinear mapping $F : \mathbb{Z}^n \to \mathbb{Z}^n$,*

$$F(\mathbf{x}) := \mathrm{rd}\,(\alpha\, H_n\, \mathbf{x})$$

with $\alpha \ge \alpha_n := \|H_n^{-1}\|_\infty$ is an invertible mapping, and we have

$$\mathbf{x} = \mathrm{rd}\,\left(\tfrac{1}{\alpha}\, H_n^{-1}\, F(\mathbf{x})\right).$$

Moreover, it follows

$$\|\alpha\,\hat{F}(\mathbf{x}) - F(\mathbf{x})\|_\infty \le \tfrac{1}{2},$$

i.e., F is close to the linear mapping $\hat{F}$.

Proof. By Theorem 2.1 we only need to show that $\alpha\, H_n$ satisfies the expansion condition

$$[-\tfrac{1}{2},\, \tfrac{1}{2})^n \subseteq \alpha\, H_n((-\tfrac{1}{2},\, \tfrac{1}{2}]^n).$$

This condition is equivalent with

$$H_n^{-1}\, \mathbf{r} \subseteq (-\tfrac{\alpha}{2},\, \tfrac{\alpha}{2}]^n \qquad \forall\, \mathbf{r} \in [-\tfrac{1}{2},\, \tfrac{1}{2})^n.$$

For $\alpha \geq \|H_n^{-1}\|_\infty$ this relation is obviously satisfied and the assertions follow from Theorem 2.1. $\square$

In the next sections, we want to solve the problem for the trigonometric matrix C_n^{II} and for matrices corresponding to periodic biorthogonal wavelet transforms, and derive simple nonlinear transforms F mapping integers to integers and keeping all features of the corresponding linear mapping $\hat{F}$.

3 Integer DCT-II of radix-2 length

Let $n \geq 2$ be a given integer. We consider the *cosine matrices* of type II with order n,

$$C_n^{II} := \sqrt{\tfrac{2}{n}} \left(\epsilon_n(j)\, \cos \tfrac{j(2k+1)\pi}{2n}\right)_{j,k=0}^{n-1}, \tag{3.1}$$

where $\epsilon_n(0) := \sqrt{2}/2$ and $\epsilon_n(j) := 1$ for $j \in \{1, \ldots, n-1\}$. Observe that C_n^{II} is orthogonal, i.e., $(C_n^{II})^{-1} = (C_n^{II})^T$ (see e.g. [30], pp. 13–14). The *discrete cosine transform of type II* (DCT-II) is a linear mapping generated by C_n^{II}. For $n := 2^t$, $t \in \mathbb{N}$, there are fast algorithms for this transform with less than $2n \log n$ arithmetical operations (see e.g. [19, 30, 22]). We want to present a simple nonlinear invertible mapping $F : \mathbb{Z}^n \to \mathbb{Z}^n$ which approximates the DCT-II suitably.

Theorem 3.1 *Let $n := 2^t$, $t \in \mathbb{N}$, be given. Then the nonlinear mapping $F : \mathbb{Z}^n \to \mathbb{Z}^n$,*

$$F(\mathbf{x}) = \mathrm{rd}\,(\alpha\, C_n^{II}\, \mathbf{x})$$

with

$$\alpha \geq \alpha_n := \tfrac{1}{\sqrt{n}} + \tfrac{1}{\sqrt{2n}}(\cot(\tfrac{\pi}{4n}) - 1)$$

is invertible, and we have

$$\mathbf{x} = \mathrm{rd}\, \left(\tfrac{1}{\alpha}(C_n^{II})^T\, F(\mathbf{x})\right).$$

Moreover, comparing $\hat{\mathbf{x}} := \alpha\, C_n^{II}\mathbf{x}$ and $\mathbf{y} = F(\mathbf{x})$ componentwisely, we find

$$|y_j - \hat{x}_j| < \tfrac{1}{2}, \qquad j = 0, \ldots, n-1,$$

i.e., in all components, the nonlinear mapping F rounds the exact (scaled) DCT-value to the next integer.

Proof. We show that

$$\|(C_n^{II})^{-1}\|_\infty = \|(C_n^{II})^T\|_\infty = \tfrac{1}{\sqrt{n}} + \tfrac{1}{\sqrt{2n}}(\cot(\tfrac{\pi}{4n}) - 1).$$

With $|(C_n^{II})^T|$ we denote the the cosine matrix, where each component is replaced by its absolute value.
Observe that for the matrix $|(C_n^{II})^T|$, the sum of all entries in each row is the same, i.e., with $\mathbf{1} := (1, \ldots, 1)^T \in \mathbb{Z}^n$ we have

$$|(C_n^{II})^T|\,\mathbf{1} = \sqrt{\tfrac{2}{n}} \left(\tfrac{\sqrt{2}}{2} + \sum_{j=1}^{n-1} \cos \tfrac{j\pi}{2n} \right) \mathbf{1}. \tag{3.2}$$

This can be seen as follows. In the k-th row ($k \in \{0, \ldots, n-1\}$) of $|(C_n^{II})^T|$ we have the values $\sqrt{\frac{2}{n}}|\cos \frac{j(2k+1)\pi}{2n}|$, $j = 0, \ldots, n-1$. Let k be fixed for a moment. Then for every $j \in \{0, \ldots, n-1\}$ there is an $j_0 \in \{0, \ldots, n-1\}$, namely $j_0 := (2k+1)j \bmod n$, such that

$$|\cos \tfrac{j(2k+1)\pi}{2n}| = \cos \tfrac{j_0\pi}{2n}.$$

Moreover, if there were two integers $j_1, j_2 \in \{0, \ldots, n-1\}$ with $(2k+1)j_1 \bmod n = (2k+1)j_2 \bmod n$, then $(2k+1)(j_1 - j_2) = 0 \bmod n$ implies $j_1 - j_2 = 0$ since $(2k+1)$ and $n = 2^t$ have no common divisor greater than 1.
Hence, by

$$\sum_{j=1}^{n-1} \cos \tfrac{j\pi}{2n} = \frac{\cos \frac{\pi}{4} \sin \frac{(n-1)\pi}{4n}}{\sin \frac{\pi}{4n}} = \tfrac{1}{2}(\cot \tfrac{\pi}{4n} - 1)$$

it follows that

$$\alpha_n \|(C_n^{II})^T\|_\infty = \sqrt{\tfrac{2}{n}} \left(\tfrac{\sqrt{2}}{2} + \sum_{j=1}^{n-1} \cos \tfrac{j\pi}{2n} \right) = \tfrac{1}{\sqrt{n}} + \tfrac{1}{\sqrt{2n}} \left(\cot \tfrac{\pi}{4n} - 1 \right). \qquad \square$$

From Theorem 3.1 we obtain the following constants for α_n for radix-2 length DCT-II algorithms.

$$\begin{array}{llll}
\alpha_2 & \approx 1.414213562, & \alpha_4 & \approx 1.923879532 \\
\alpha_8 & \approx 2.641845989, & \alpha_{16} & \approx 3.671595601 \\
\alpha_{32} & \approx 5.143712179, & \alpha_{64} & \approx 7.238780613 \\
\alpha_{128} & \approx 10.21167688, & \alpha_{256} & \approx 14.42332168 \\
\alpha_{512} & \approx 20.38476090, & \alpha_{1024} & \approx 28.81926938
\end{array}$$

In particular, the constants α_n satisfy $\alpha_n \le \sqrt{2}^t = \sqrt{n}$, i.e., we can replace the normalization factor $\sqrt{2/n}$ in the definition of C_n by $\sqrt{2}$ and apply a fast DCT algorithm to this scaled DCT. Based on the above observations we propose the following Algorithm for an integer DCT-II.

Algorithm 3.2 [Integer DCT–II algorithm]
Input: $\mathbf{x} \in \mathbb{Z}^n$ where $n := 2^t$.

1. Compute $\hat{\mathbf{x}} := \alpha\, C_n^{II}\mathbf{x}$ by a fast DCT-II algorithm, where $\alpha \ge \alpha_n$ is chosen suitably. For example, take $\alpha := \sqrt{n}$.

2. Compute $\mathbf{y} := \mathrm{rd}(\hat{\mathbf{x}})$.

Output: $\mathbf{y} \in \mathbb{Z}^n$ approximating $\alpha C_n^{II}\mathbf{x}$.

The above algorithm simply uses the well-known fast DCT-algorithms, being implemented already. For example, for $n = 8$ one can just take the factor $\alpha = 2\sqrt{2} > \alpha_n$, and use a fast algorithm for $2\sqrt{2}\, C_8^{II}\mathbf{x}$, which can be done e.g. by 11 multiplications and 29 additions (see [19]).
The inverse algorithm is equivalently simple.

Algorithm 3.3 [Inverse Integer DCT–II algorithm]
Input: $\quad \mathbf{y} \in \mathbb{Z}^n$ where $n := 2^t$.

1. Compute $\hat{\mathbf{y}} := \frac{1}{\alpha}\,(C_n^{II})^T\mathbf{y}$ by a fast inverse DCT-II algorithm, where α is chosen as in Algorithm 3.2.

2. Compute $\mathbf{x} := \mathrm{rd}\,(\hat{\mathbf{y}})$.

Output: $\quad \mathbf{x} \in \mathbb{Z}^n$ original input vector of Algorithm 3.2.

Remarks.
1. If one wants to use fixed point arithmetic, the irrational coefficients used in the fast DCT-II algorithm need to be replaced by dyadic rationals (see e.g. [29]). For example, using the fast split-radix algorithm for the DCT-II of length $n = 8$ proposed in [22] and lifting, the values

$$\sqrt{2},\ \tan\tfrac{\pi}{32},\ \tan\tfrac{3\pi}{32},\ \tan\tfrac{\pi}{16},\ \tan\tfrac{\pi}{8},\ \sin\tfrac{\pi}{16},\ \sin\tfrac{3\pi}{16},\ \sin\tfrac{\pi}{8},\ \sin\tfrac{\pi}{4}$$

need to be approximated (see e.g. [24]). For the obtained transformation matrix $\tilde{C}_n^{II}$, Theorem 2.2 can be applied as before.

2. For the two-dimensional DCT-II, which is frequently used in image compression, one can use the row-column method. In this way one obtains results which are very close to the usual method of image compression, where the entries of the resulting matrix $\hat{X} = 8\,C_8^{II}\,X\,(C_8^{II})^T$ are rounded to the next integers.

3. Using Theorem 2.2, integer algorithms can be derived for other trigonometric transforms analogously.

4 Integer wavelet algorithms

Biorthogonal wavelets are given by two dual sets of coefficients, the analysis filters $\{h_k\}$, $\{g_k\}$ and the synthesis filters $\{\tilde{h}_k\}$, $\{\tilde{g}_k\}$. We assume that these sequences are real and have finite length. Let us further assume that the low-pass filters satisfy

$$\sum_{k\in} h_k = a, \qquad \sum_{k\in} \tilde{h}_k = \tfrac{2}{a}, \tag{4.1}$$

where $a \in \mathbb{R}$, $a > 0$ is a normalization constant. For given low-pass filters $h = \{h_k\}$ and $\tilde{h} = \{\tilde{h}_k\}$, the filters $g = \{g_k\}$ and $\tilde{g} = \{\tilde{g}_k\}$ are given by

$$g_k := a'\,(-1)^{k-1}\tilde{h}_{1-k}, \qquad \tilde{g}_k := \tfrac{1}{a'}\,(-1)^{k-1}h_{1-k} \tag{4.2}$$

with $a' \in \mathbb{R}$. For $a = \sqrt{2}$, $a' = 1$, we say that the filters h, $\tilde{h}$, g, and $\tilde{g}$ are *normalized*. For applications, one often likes to have filter coefficients which are dyadic rationals in order to use fixed-point arithmetic. For a lot of biorthogonal wavelet filter banks this can be obtained indeed by taking dyadic numbers for a and a' (see the examples in Subsection 4.2).

4.1 Periodic wavelet transform

One of the simplest techniques for handling finite-extent signals is the periodic extension. Note that the periodic extension transform can be viewed as a linear transform on $\mathbb{R}^n$. For implementations, it is conceptually simpler to apply a fixed wavelet filter bank to the periodically extended signal. This perspective is taken in the applications (see Subsection 4.2).

For the theoretical treatment, in this subsection we shall use the other perspective, namely the periodization of the filters, which are then applied to a finite length signal.

Let us consider the periodic wavelet transform. Let $N_0 \in \mathbb{N}$ and $j_0 \in \mathbb{N}$ be fixed and let $n := 2^{j_0} N_0$ and $n_j := 2^{-j} n$ for $j = 0, \ldots, j_0$. We form the periodic filters

$$\begin{aligned} h_{j,k} &:= \sum_{l=-\infty}^{\infty} h_{k+n_j l}, \quad \tilde{h}_{j,k} := \sum_{l=-\infty}^{\infty} \tilde{h}_{k+n_j l}, \\ g_{j,k} &:= \sum_{l=-\infty}^{\infty} g_{k+n_j l}, \quad \tilde{g}_{j,k} := \sum_{l=-\infty}^{\infty} \tilde{g}_{k+n_j l}. \end{aligned} \tag{4.3}$$

If n_j is greater than the length of the filters h, $\tilde{h}$, then these series contain only one nontrivial summand. Let

$$\begin{aligned} H_j &:= (h_{j,r-2k})_{r,k=0}^{n_j-1,n_j/2-1}, \qquad \tilde{H}_j := (\tilde{h}_{j,r-2k})_{r,k=0}^{n_j-1,n_j/2-1}, \\ G_j &:= (g_{j,r-2k})_{r,k=0}^{n_j-1,n_j/2-1}, \qquad \tilde{G}_j := (\tilde{g}_{j,r-2k})_{r,k=0}^{n_j-1,n_j/2-1} \end{aligned}$$

be the corresponding matrices, such that

$$\tilde{H}_j\, H_j^T + \tilde{G}_j\, G_j^T = I_{n_j},$$

where I_{n_j} denotes the identity matrix of length n_j.

For a given vector $\mathbf{s}^0 \in \mathbb{R}^n$, the periodic discrete wavelet transform (or wavelet decomposition) of $\mathbf{s}^0$ through L levels ($L \le j_0$) is given by $\mathbf{w}^L := ((\mathbf{s}^L)^T, (\mathbf{d}^L)^T, (\mathbf{d}^{L-1})^T, \ldots, (\mathbf{d}^1)^T)^T$, where we compute iteratively

$$\mathbf{s}^{m+1} := H_m^T\, \mathbf{s}^m, \qquad \mathbf{d}^{m+1} := G_m^T\, \mathbf{s}^m, \qquad m = 0, \ldots, L-1.$$

Here the vectors $\mathbf{s}^m$ and $\mathbf{d}^m$ have length $n_m = 2^{-m} n$ for $m = 1, \ldots, L$. The matrix multiplications are equivalent with computing the (periodic) convolutions

$$s_k^{m+1} = \sum_{r=0}^{n_m-1} h_{m,r-2k}\, s_r^m, \qquad d_k^{m+1} = \sum_{r=0}^{n_m-1} g_{m,r-2k}\, s_r^m$$

for $k = 0, \ldots, n_{m-1} - 1$, where $\mathbf{s}^{m+1} := (s_k^{m+1})_{k=0}^{n_m-1}$, $\mathbf{d}^{m+1} := (d_k^{m+1})_{k=0}^{n_m-1}$.

The periodic inverse discrete wavelet transform (or wavelet reconstruction) is based on

$$\mathbf{s}^m := \tilde{H}_m\, \mathbf{s}^{m+1} + \tilde{G}_m\, \mathbf{d}^{m+1}$$

or equivalently,

$$s_r^m := \sum_{k=0}^{n_m-1} \tilde{h}_{m,r-2k}\, s_k^{m+1} + \tilde{g}_{m,r-2k}\, d_k^{m+1}, \qquad r = 0, \ldots, n_m - 1.$$

We define the direct sum of two matrices A and B as a block matrix $A \oplus B := \mathrm{diag}\,(A,\, B)$. Further, let

$$M_j := \begin{pmatrix} H_j^T \\ G_j^T \end{pmatrix} \in \mathbb{R}^{n_j \times n_j}.$$

Then the matrix representation of the periodic discrete wavelet transform has the form

$$\mathbf{w}^L := (M_{L-1} \oplus I_{n-n_{L-1}})(M_{L-2} \oplus I_{n-n_{L-2}}) \ldots (M_1 \oplus I_{n-n_1})\, M_0\, \mathbf{s}^0.$$

That is, setting for fixed $n := 2^{j_0} N_0$ and $L \in \{1, \ldots, j_0\}$,

$$H_{n,L} := (M_{L-1} \oplus I_{n-n_{L-1}})(M_{L-2} \oplus I_{n-n_{L-2}}) \ldots (M_1 \oplus I_{n-n_1})\, M_0,$$

the periodic discrete wavelet transform through L levels can be written as the linear mapping

$$\mathbf{w}^L = H_{n,L}\, \mathbf{s}^0.$$

As for the DCT, we now use Theorem 2.2 to obtain an integer wavelet transform. Hence we need to compute $\alpha_{n,L} := \|H_{n,L}^{-1}\|_\infty$. Observe that

$$H_{n,L}^{-1} = M_0^{-1}\, (M_1^{-1} \oplus I_{n-n_1}) \ldots (M_{L-1}^{-1} \oplus I_{n-n_{L-1}})$$

with $M_j^{-1} = (\tilde{H}_j,\, \tilde{G}_j)^T$.
The algorithms for the integer wavelet transform now work formally as before.

Algorithm 4.1 [Integer wavelet algorithm]
Input: $\mathbf{s}^0 \in \mathbb{Z}^n$ where $n = N_0\, 2^{j_0}$.
L (number of levels for wavelet decomposition)

1. Compute $\mathbf{w}^L := \alpha\, H_{n,L}\, \mathbf{s}^0$ by the fast wavelet transform, where $\alpha \geq \alpha_{n,L}$ is chosen suitably.
2. Compute the integer approximation $\mathbf{y} := \mathrm{rd}\,(\mathbf{w}^L)$.

Output: $\mathbf{y} \in \mathbb{Z}^n$ approximating $\mathbf{w}^L$.

Algorithm 4.2 [Inverse Integer wavelet algorithm]
Input: $\mathbf{y} \in \mathbb{Z}^n$ where where $n = N_0\, 2^{j_0}$.
L (number of levels for wavelet reconstruction, as in Algorithm 4.1)

1. Compute $\mathbf{v} := \frac{1}{\alpha}\, H_{n,L}^{-1}\, \mathbf{y}$ by the inverse fast wavelet transform, where α is chosen as in Algorithm 4.1.
2. Put $\mathbf{s}^0 := \mathrm{rd}\,(\mathbf{v})$.

Output: $\mathbf{s}^0 \in \mathbb{Z}^n$ original input vector of Algorithm 4.1.

For the realization of the fast wavelet transform (and the inverse wavelet transform) we can apply periodic extension of the signal and the lifting method (see Subsection 4.2).
In order to determine $\alpha_{n,L}$, let us consider the structure of $H_{n,L}^{-1}$ in more detail. Obviously, $H_{n,1}^{-1} = (\tilde{H}_0, \tilde{G}_0)$. Let

$$\tilde{h}(z) = \sum_{l\in} \tilde{h}_l z^l, \qquad \tilde{g}(z) = \sum_{l\in} \tilde{g}_l z^l$$

be the z-transforms of the synthesis filters $\tilde{h}$ and $\tilde{g}$. We introduce the filters $\tilde{h}^1$, $\tilde{g}^1$ by

$$\tilde{h}^1(z) := \tilde{h}(z)\,\tilde{h}(z^2), \qquad \tilde{g}^1(z) := \tilde{h}(z)\,\tilde{g}(z^2).$$

Further, let $\tilde{h}_0^1$, $\tilde{g}_0^1$ be the n-periodizations of $\tilde{h}_0^1$ and $\tilde{g}_0^1$ as in (4.3). Then a comparison of matrix entries leads to

$$H_{n,2}^{-1} = (\tilde{H}_0\tilde{H}_1,\ \tilde{H}_0\tilde{G}_1,\ \tilde{G}_0) = (\tilde{H}_0^1,\ \tilde{G}_0^1, \tilde{G}_0),$$

where

$$\tilde{H}_0^1 := (\tilde{h}_{0,r-4k}^1)_{r,k=0}^{n-1,n/4-1}, \qquad \tilde{G}_0^1 := (\tilde{g}_{0,r-4k}^1)_{r,k=0}^{n-1,n/4-1}.$$

Generally, with

$$\tilde{h}^m(z) := \tilde{h}(z)\,\tilde{h}(z^2)\ldots\tilde{h}(z^{2^m}), \qquad \tilde{g}^m(z) := \tilde{h}(z)\ldots\tilde{h}(z^{2^{m-1}})\tilde{g}(z^{2^m}) \tag{4.4}$$

for $m = 1, \ldots, L-1$, and their n-periodizations $\tilde{h}_0^m$, $\tilde{g}_0^m$ (defined analogously as in (4.3)) we find

$$H_{n,L}^{-1} = (\tilde{H}_0^{L-1},\ \tilde{G}_0^{L-1}, \ldots, \tilde{G}_0^1,\ \tilde{G}_0),$$

where

$$\tilde{H}_0^m := (\tilde{h}_{0,r-2^{m+1}k}^m)_{r,k=0}^{n-1,2^{-m-1}n-1}, \qquad \tilde{G}_0^m := (\tilde{g}_{0,r-2^{m+1}k}^m)_{r,k=0}^{n-1,2^{-m-1}n-1}.$$

We obtain the following

Theorem 4.3 *Let finite biorthogonal pairs of analysis filters h, g and synthesis filters $\tilde{h}$, $\tilde{g}$ be given. Let the filters $\tilde{h}^m$ and $\tilde{g}^m$ be defined as in (4.4) and $n := 2^{j_0}N_0$. Then, the periodic integer wavelet transform through L ($L \le j_0$) levels $F : \mathbb{Z}^n \to \mathbb{Z}^n$,*

$$F(\mathbf{s}^0) := \mathrm{rd}\,(\alpha\, H_{n,L}\,\mathbf{s}^0),$$

with the expansion factor

$$\alpha \ge \alpha_{n,L} := \|H_{n,L}^{-1}\|_\infty = \max_{k=0,\ldots,2^L-1}\left(\sum_{r\in} |\tilde{h}_{0,k+2^L r}^{L-1}| + \sum_{\nu=0}^{L-1}\sum_{r\in} |\tilde{g}_{0,k+2^{\nu+1}r}^{\nu}|\right)$$

is invertible, and we have

$$\mathbf{s}^0 = \mathrm{rd}\,(\tfrac{1}{\alpha} H_{n,L}^{-1}\, F(\mathbf{s}^0)).$$

Proof. Considering the matrix $|H_{n,L}^{-1}|$, which contains only the absolute values of the entries of $H_{n,L}^{-1}$, we see that the sum of the entries in the k-th row, $0 \le k \le n-1$, is

$$\sum_{r\in} |\tilde{h}_{0,k+2^L r}^{L-1}| + \sum_{\nu=0}^{L-1}\sum_{r\in} |\tilde{g}_{0,k+2^{\nu+1}r}^{\nu}|.$$

Further, the structure of $H_{n,L}^{-1}$ implies that the sum of entries in the $(k + j\,2^L)$-th row coincides with the sum in the k-th row for $j \in \mathbb{Z}$ and $0 \le k + j2^L \le n-1$. Hence, we find $\alpha_{n,L} = \|H_{n,L}^{-1}\|_\infty$ and the assertions follow from Theorem 2.2. □

Remarks.
1. Observe that $\tilde{h}^{L-1}_{0,k+2^L r} = \tilde{h}^{L-1}_{k+2^L r}$ and $\tilde{g}^{\nu}_{0,k+2^{\nu+1}r} = \tilde{g}^{\nu}_{k+2^{\nu+1}r}$ for $\nu = 0,\ldots,L-1$ if we assume that n is greater than the number of nonzero coefficients in $\tilde{h}^{L-1}$ and in $\tilde{g}^{L-1}$. Indeed, one simply observes that the constant α_L given by

$$\alpha_L := \max_{k=0,\ldots,2^L-1} \left(\sum_{r\in} |\tilde{h}^{L-1}_{k+2^L r}| + \sum_{\nu=0}^{L-1} \sum_{r\in} |\tilde{g}^{\nu}_{k+2^{\nu+1}r}| \right), \tag{4.5}$$

either coincides with $\alpha_{n,L}$ in Theorem 4.3 (if n is great enough) or may be (slightly) greater than $\alpha_{n,L}$. For the computation of α_L, we need not to consider the periodizations of the filter. In the examples in Subsection 3.2 we will compute the constants α_L, which are independent of n.

2. Simple periodization of a signal often leads to large jumps at the splice points between periods and may introduce additional high frequency content. In order to moderate these effects, the **symmetric extension** is used frequently. With this technique, a signal is extended, so that it is both, symmetric and periodic. There are different ways to extend a signal symmetrically (see e.g. [3, 28]). Symmetric extension is usually employed with biorthogonal filter banks which preserve symmetry, in order to be nonexpansive. Observe that the kind of extension must be chosen carefully depending on the filter bank in order to obtain symmetric subbands again.
Taking a suitable symmetric extension of $\mathbf{s}^0$, all observations in this section can be applied as before.

4.2 Application to biorthogonal filter banks

In this subsection we will apply the method to different orthogonal and biorthogonal filter banks. In particular, we shall compute the expansion factors α_L needed for the integer wavelet transformation through L levels. Further, we study the relevance of the normalization factors a and a' in (4.1) and (4.2).

For biorthogonal filter banks we shall consider the *normalized case* $a = \sqrt{2}$, $a' = 1$, the *downward normalization* $a = 1$, $a' = 1$ taken in all decomposition levels analogously as in [4], and the *alternating normalization*, where we take the normalization factors $a = 2$, $a' = 2$ in odd decomposition levels and $a = 1$, $a' = \frac{1}{2}$ in even decomposition levels. Observe, that for different normalizations we obtain different expansion factors α_L. Assume that the transformation matrix $H_{n,L}$ is computed from the filters in the normalized case. Then the *downward normalization* corresponds to the transformation matrix

$$\left((\sqrt{2})^{-L} I_{n_L} \oplus (\sqrt{2})^{-L+2} I_{n_L} \oplus (\sqrt{2})^{-L+3} I_{n_{L-1}} \ldots \oplus (\sqrt{2})^{-1} I_{n_3} \oplus I_{n_2} \oplus \sqrt{2} I_{n_1}\right) H_{n,L}$$

with $n_j := 2^{-j} n$, i.e., compared with the linear normalized wavelet transform through L levels, $\mathbf{s}^L$ will be multiplied with $\alpha_L\,(\sqrt{2})^{-L}$ and $\mathbf{d}^j$ with $2\,\alpha_L\,(\sqrt{2})^{-j}$ in the integer wavelet

algorithm before rounding off. This downward normalization may be especially interesting for the application to signals with small content in the high frequency subbands.
Alternating normalization corresponds to the transformation matrix

$$\left(I_{n_{L-1}} \oplus \sqrt{2} I_{n_{L-1}} \oplus I_{n_{L-2}} \oplus \ldots \oplus I_{n_2} \oplus \sqrt{2} I_{n_1}\right) H_{n,L}$$

for odd L and

$$\left(\sqrt{2}\, I_{n_{L-1}} \oplus I_{n_{L-1}} \oplus \sqrt{2}\, I_{n_{L-2}} \oplus \ldots \oplus I_{n_2} \oplus \sqrt{2} I_{n_1}\right) H_{n,L}$$

for even L. Hence, compared with the normalized linear wavelet transform through L levels, $\mathbf{d}^j$ will be multiplied with $\alpha_L \sqrt{2}$ for odd j and with α_L for even j in the integer wavelet algorithm, and $\mathbf{s}^L$ has the same factor as $\mathbf{d}^L$. Naturally, the constants α_L are much smaller for alternating normalization then for downward normalization. In our examples, the alternating and the downward normalization lead to filter coefficients which are dyadic rationals in each step, such that we can apply fixed point arithmetic, while in the normalized case the computations involve irrational filter coefficients.
In the examples below, we want to show, that again the lifting method can be used to perform the fast wavelet transform, but compared with the the known integer wavelet transforms based on lifting steps and rounding off, the intermediate rounding operations are dropped, such that we stay to be very close to the linear wavelet transform (scaled by the expansion factor α_L).

Biorthogonal $(2,2)$ interpolation transform.

We want to explain the procedure extensively for the biorthogonal $(2,2)$ interpolating transform (see e.g. [9]). The *normalized* coefficients of this filter bank ($a = \sqrt{2}$, $a' = 1$) are given by the analysis filters

$$\begin{aligned} h_{-2} &:= \tfrac{-\sqrt{2}}{8},\ g_{-1} := \tfrac{2\sqrt{2}}{8},\ h_0 := \tfrac{6\sqrt{2}}{8},\ h_1 := \tfrac{2\sqrt{2}}{8},\ h_2 := \tfrac{-\sqrt{2}}{8} \\ g_0 &:= \tfrac{-\sqrt{2}}{4},\ g_1 := \tfrac{2\sqrt{2}}{4},\ g_2 := \tfrac{-\sqrt{2}}{4}, \end{aligned} \tag{4.6}$$

and the synthesis filters follow from (4.2).
First we consider the periodic discrete wavelet transform through 1 level for this normalized filters and obtain by Theorem 4.3 that

$$\alpha_1 = \max\{|\tilde{h}_0| + |\tilde{g}_0| + |\tilde{g}_2|, |\tilde{h}_{-1}| + |\tilde{h}_1| + |\tilde{g}_{-1}| + |\tilde{g}_1| + |\tilde{g}_3|\} = \tfrac{3\sqrt{2}}{2}.$$

For the computation of $\alpha_1\, H_{n,1}\, \mathbf{s}^0$ (normalized case) we use the lifting method:
Compute

$$\begin{aligned} d_k^1 &:= \tfrac{3}{2}\left(s^0_{(2k+1)\mathrm{mod}\ n} - \tfrac{1}{2}(s^0_{2k\mathrm{mod}\ n} + s^0_{(2k+2)\mathrm{mod}\ n})\right), && k = 0, \ldots, \tfrac{n}{2} - 1, \\ s_k^1 &:= 3\, s^0_{2k\mathrm{mod}\ n} + \tfrac{1}{2}(d^1_{(k-1)\mathrm{mod}\ n/2} + d^1_{k\mathrm{mod}\ n/2}), && k = 0, \ldots, \tfrac{n}{2} - 1 \end{aligned}$$

and put $\mathbf{y}^1 := \mathrm{rd}(\mathbf{w}^1)^T = \mathrm{rd}((\mathbf{s}^1)^T, (\mathbf{d}^1)^T)^T \in \mathbb{Z}^n$, where $\mathbf{s}^1 := (s_k^1)_{k=0}^{n/2-1}$, $\mathbf{d}^1 := (d_k^1)_{k=0}^{n/2-1}$.
The inverse integer wavelet transform $\frac{1}{\alpha_1} H_{n,1}^{-1}\, \mathbf{y}^1$ with $\alpha_1 = \frac{3\sqrt{2}}{2}$ can be computed by

$$\begin{aligned} \hat{s}^0_{2k} &:= s^1_{k\mathrm{mod}\ n/2} - \tfrac{1}{2}(d^1_{k-1\mathrm{mod}\ n/2} + d^1_{k\mathrm{mod}\ n/2}), && k = 0, \ldots, \tfrac{n}{2} - 1, \\ \hat{s}^0_{2k+1} &:= 2\, d^1_{k\mathrm{mod}\ n/2} + \tfrac{1}{2}(\hat{s}^0_{2k\mathrm{mod}\ n} + \hat{s}^0_{2k+2\mathrm{mod}\ n}) && k = 0, \ldots, \tfrac{n}{2} - 1. \end{aligned}$$

After rounding off we obtain the original vector $\mathbf{s}^0 = \mathrm{rd}(\frac{1}{3}\,\hat{\mathbf{s}}^0)$.
If one wants avoid the division by 3, one can take $\alpha = 2\sqrt{2}$ instead of $\alpha = \frac{3}{2}\sqrt{2}$ (enlarging the resulting coefficients further).
For the downward normalization $a = 1$, $a' = 1$, the filter coefficients h_k in (4.6) are divided by $\sqrt{2}$ and the g_k are multiplied with $\sqrt{2}$. In this case we find the expansion factor $\alpha_1 = 2$, and

$$\begin{aligned} d_k^1 &:= \alpha_1 \left(s^0_{(2k+1)\mathrm{mod}\ n} - \tfrac{1}{2}(s^0_{2k\mathrm{mod}\ n} + s^0_{(2k+2)\mathrm{mod}\ n}) \right), & k = 0, \ldots, \tfrac{n}{2} - 1, \\ s_k^1 &:= \alpha_1\, s^0_{2k\mathrm{mod}\ n} + \tfrac{1}{4}(d^1_{(k-1)\mathrm{mod}\ n/2} + d^1_{k\mathrm{mod}\ n/2}), & k = 0, \ldots, \tfrac{n}{2} - 1, \end{aligned}$$

followed by rounding off. Here, the coefficients in $\mathbf{s}^1$ and $\mathbf{d}^1$ are multiplied with $\sqrt{2}$ and $2\sqrt{2}$, respectively (compared with the linear normalized transform). Observe, that this procedure is similar to the integer lifting transform in [4], formula (4.1), but not the same. Here, we have no rounding off after the first step and have to pay for being closer to the linear transform with the expansion factor $\alpha_1 = 2$.
Alternating normalization leads for $L = 1$ in this example to the same algorithm as the normalized case.

Let us now consider the periodic discrete wavelet transform through L levels with $L \geq 2$.
1. Normalized case. In the case of *normalized* filters ($a = \sqrt{2}$, $a' = 1$), we find for $L = 2$ with

$$\begin{aligned} \tilde{h}^1(z) &= \tilde{h}(z)\tilde{h}(z^2) = \tfrac{1}{8}(z^{-3} + 2z^{-2} + 3z^{-1} + 4 + 3z + 2z^2 + z^3) \\ \tilde{g}^1(z) &= \tilde{h}(z)\tilde{g}(z^2) = \tfrac{1}{16}(-\tfrac{1}{z^3} - \tfrac{2}{z^2} - \tfrac{3}{z} - 4 + 4z + 12z^2 + 4z^3 - 4z^4 - 3z^5 - 2z^6 - z^7) \end{aligned}$$

the expansion factor

$$\alpha_2 = \max_{j \in \{0,1,2,3\}} \{ \sum_{l \in} |\tilde{h}^1_{j+4l}| + |\tilde{g}^1_{j+4l}| + |\tilde{g}_{j+2l}| \} = 1 + \sqrt{2}.$$

For $L \geq 2$, the expansion factors α_L are given numerically in Table 1. One obtains now the following algorithm. Let $\mathbf{s}^0 \in \mathbb{Z}^n$, $n_j := n/2^j$ and choose $\alpha \geq \alpha_L$.
For j from 1 to L do

$$\begin{aligned} d_k^j &:= \tfrac{\sqrt{2}}{2} \left(s^{j-1}_{(2k+1)\mathrm{mod}\ n_{j-1}} - \tfrac{1}{2}(s^{j-1}_{(2k+2)\mathrm{mod}\ n_{j-1}} + s^{j-1}_{2k\mathrm{mod}\ n_{j-1}}) \right), & k = 0, \ldots, n_j - 1, \\ s_k^j &:= \sqrt{2}\, s^{j-1}_{2k\mathrm{mod}\ n_{j-1}} + \tfrac{1}{2}(d^j_{(k-1)\mathrm{mod}\ n_j} + d^j_{k\mathrm{mod}\ n_j}), & k = 0, \ldots, n_j - 1, \end{aligned}$$

and put $\mathbf{y}^L := \mathrm{rd}\,\alpha\,((\mathbf{s}^L)^T, (\mathbf{d}^L)^T, \ldots, (\mathbf{d}^1)^T)^T$.
Compared with the normalized linear wavelet transform, the coefficients in $\mathbf{s}^L$, and $\mathbf{d}^j$ and are multiplied with $\alpha \geq \alpha_L$. Unfortunately, the above algorithm is not appropriate for fixed-point arithmetic, and rational coefficients can not be obtained just by a suitable choice of α. The inverse algorithm works as before, going back step by step and rounding off at last.

2. Downward normalization. We consider the *downward normalization* ($a = 1$, $a' = 1$) used in [4]. For $L = 2$ this corresponds to the transformation matrix $(\frac{1}{2}I_{n/4} \oplus I_{n/4} \oplus \sqrt{2}I_{n/2})\,H_{n,2}$. In this case we obtain the expansion factor $\alpha_2 = \frac{5}{2}$. For $L \geq 2$ the exact constants are given in Table 1. The algorithm reads for given $\mathbf{s}^0 \in \mathbb{Z}^n$ and $\alpha \geq \alpha_L$:

For j from 1 to L do

$$\begin{aligned} d_k^j &:= s^{j-1}_{(2k+1)\mathrm{mod}\ n_{j-1}} - \tfrac{1}{2}(s^{j-1}_{(2k+2)\mathrm{mod}\ n_{j-1}} + s^{j-1}_{2k\mathrm{mod}\ n_{j-1}}), & k = 0,\ldots,n_j-1,\\ s_k^j &:= s^{j-1}_{2k\mathrm{mod}\ n_{j-1}} + \tfrac{1}{4}(d^j_{(k-1)\mathrm{mod}\ n_j} + d^j_{k\mathrm{mod}\ n_j}), & k = 0,\ldots,n_j-1,\end{aligned}$$

and put $\mathbf{y}^L := \mathrm{rd}\,\alpha\,((\mathbf{s}^L)^T,\, (\mathbf{d}^L)^T, \ldots, (\mathbf{d}^1)^T)^T$.

Comparing with the normalized linear wavelet transform, we find in the case $L = 2$ the factors $\frac{5}{4}$ for $\mathbf{s}^2$, $\frac{5}{2}$ for $\mathbf{d}^2$ and $\frac{5\sqrt{2}}{2}$ for $\mathbf{d}^1$.

3. Alternating normalization. The *alternating normalization* is also suitable for fixed-point arithmetic. In the first transformation level, we take the normalization factors $a = 2$, $a' = 2$ in (4.1) and (4.2), i.e., the filter coefficients h_k and g_k in (4.6) are multiplied with $\sqrt{2}$. In the second transformation level, we use the normalization factors $a = 1$, $a' = \frac{1}{2}$ in (4.1) and (4.2). In this case, we obtain for $L = 2$ the expansion factor $\alpha_2 = 2$. For $L \geq 2$ the exact expansion factors α_L are in Table 1. For even $L = 2l$, $\alpha \geq \alpha_L$ and given $\mathbf{s}^0 \in \mathbb{Z}^n$ the algorithm reads as follows:
For j from 1 to l do

$$\begin{aligned} d_k^{2j-1} &:= s^{2j-2}_{(2k+1)\mathrm{mod}\ n_{2j-2}} - \tfrac{1}{2}(s^{2j-2}_{(2k+2)\mathrm{mod}\ n_{2j-2}} + s^{2j-2}_{2k\mathrm{mod}\ n_{2j-2}}), & k = 0,\ldots,n_{2j-1}-1,\\ s_k^{2j-1} &:= 2\, s^{2j-2}_{2k\mathrm{mod}\ n_{2j-2}} + \tfrac{1}{2}(d^{2j-1}_{(k-1)\mathrm{mod}\ n_{2j-1}} + d^{2j-1}_{k\mathrm{mod}\ n_{2j-1}}), & k = 0,\ldots,n_{2j-1}-1,\\ d_k^{2j} &:= \tfrac{1}{2}\left(s^{2j-1}_{(2k+1)\mathrm{mod}\ n_{2j-1}} - \tfrac{1}{2}(s^{2j-1}_{(2k+2)\mathrm{mod}\ n_{2j-1}} + s^{2j-1}_{2k\mathrm{mod}\ n_{2j-1}})\right), & k = 0,\ldots,n_{2j}-1,\\ s_k^{2j} &:= s^{2j-1}_{2k\mathrm{mod}\ n_{2j-1}} + \tfrac{1}{2}(d^{2j}_{(k-1)\mathrm{mod}\ n_{2j}} + d^{2j}_{k\mathrm{mod}\ n_{2j}}), & k = 0,\ldots,n_{2j}-1,\end{aligned}$$

and put $\mathbf{w}^L := \mathrm{rd}\,\alpha\,((\mathbf{s}^L)^T,\, (\mathbf{d}^L)^T, \ldots, (\mathbf{d}^1)^T)^T$.
For odd L the procedure follows analogously.
Compared with the normalized linear wavelet transform, in the case $L = 2$ the coefficients in $\mathbf{d}^1$ are multiplied with $2\sqrt{2}$, and the coefficients in $\mathbf{d}^2$ and $\mathbf{s}^2$ with 2. In the table, we give also the size n for which n (divisible by 2^L) is greater that the filter length of $\tilde{h}^{L-1}$ and $\tilde{g}^{L-1}$.

level L	normalized case	alternating case	downward case	n
	α_L	α_L	α_L	
1	2.1213203	1.5	2.0	$n \geq 6$
2	2.4142136	2.0	2.5	$n \geq 12$
3	2.7980970	2.125	3.25	$n \geq 24$
4	3.0070436	2.4375	3.875	$n \geq 48$
5	3.1768883	2.484375	4.5625	$n \geq 96$
6	3.2891741	2.6484375	5.21875	$n \geq 192$
7	3.3713343	2.669921875	5.890625	$n \geq 384$
8	3.4284538	2.7529296875	6.5546875	$n \geq 768$
9	3.4691886	2.763427734375	7.222656625	$n \geq 1536$
10	3.4978704	2.8050537109375	7.888671875	$n \geq 3072$

Table 1. Constants α_L for biorthogonal $(2, 2)$ interpolatory filter bank

Let us consider some further examples shortly.

Biorthogonal $(4,2)$ interpolating wavelet transform (see [4, 9]).
With the coefficients

$$\begin{aligned} h_{-4} &:= \tfrac{\sqrt{2}}{64},\ h_{-3} := 0,\ h_{-2} := -\tfrac{8\sqrt{2}}{64},\ h_{-1} := \tfrac{16\sqrt{2}}{64},\ h_0 := \tfrac{46\sqrt{2}}{64},\ h_1 := \tfrac{16\sqrt{2}}{64}, \\ h_2 &:= -\tfrac{8\sqrt{2}}{64},\ h_3 := 0,\ h_4 := \tfrac{\sqrt{2}}{64}, \\ g_{-4} &:= \tfrac{\sqrt{2}}{32},\ g_{-3} := 0,\ g_{-2} := -\tfrac{9\sqrt{2}}{32},\ g_{-1} := \tfrac{16\sqrt{2}}{32}, g_0 := -\tfrac{9\sqrt{2}}{32},\ g_1 := 0,\ g_2 := \tfrac{\sqrt{2}}{32} \end{aligned}$$

we obtain the constants α_L in Table 2. Further, n in the last column of the table indicates for which n the filter lengths of $\tilde{h}^{L-1}$ and $\tilde{g}^{L-1}$ are smaller than n. For downward normalization, the lifting scheme is analogous to that in [4], formula (4.2). For alternating normalization we find in the even decomposition step

$$\begin{aligned} d_k^{2j} &:= \tfrac{1}{2}\, s^{2j-1}_{(2k-1)\mathrm{mod}\ n_{2j-1}} + \tfrac{1}{32}\, \Big((s^{2j-1}_{(2k+2)\mathrm{mod}\ n_{2j-1}} + s^{2j-1}_{(2k-4)\mathrm{mod}\ n_{2j-1}}) \\ &\qquad -9(s^{2j-1}_{2k\mathrm{mod}\ n_{2j-1}} + s^{2j-1}_{(2k-2)\mathrm{mod}\ n_{2j-1}}) \Big), \\ s_k^{2j} &:= s^{2j-1}_{2k\mathrm{mod}\ n_{2j-1}} + \tfrac{1}{2}\, (d^{2j}_{k\mathrm{mod}\ n_{2j}} + d^{2j}_{(k+1)\mathrm{mod}\ n_{2j}}), \end{aligned}$$

both for $k = 0, \ldots, n_{2j} - 1$.
In the odd decomposition step a multiplication with 2 is necessary. In the downward case and the alternating case the constants α_L in Table 2 are exact for $L \leq 3$.

level L	normalized case	alternating case	downward case	n
	α_L	α_L	α_L	
1	2.2980970	1.625	2.25	$n \geq 10$
2	2.6446823	2.23046875	2.8359375	$n \geq 24$
3	3.0987516	2.37420654296875	3.7467041015625	$n \geq 56$
4	3.3467352	2.751277923583984375	4.49417877197265625	$n \geq 112$
5	3.5481054	2.803365826606750488	5.32304525375366211	$n \geq 224$
6	3.6813222	3.000839076993624496	6.11113217473030090	$n \geq 448$
7	3.7787692	3.024440605426207185	6.91959074698388577	$n \geq 896$
8	3.8948449	3.124281642769346945	7.71785897866357118	$n \geq 1792$
9	3.9288674	3.135778127831599704	8.52122122266882798	$n \geq 3584$
10	3.9288674	3.185836284569944610	9.32203617065579238	$n \geq 7168$

Table 2. Constants α_L for the biorthogonal $(4,2)$ interpolatory transform.

Biorthogonal $(4,4)$ interpolating wavelet transform (see [9]).
With the normalized coefficients

$$\begin{aligned} h_{-6} &:= -\tfrac{\sqrt{2}}{512},\ h_{-5} := 0,\ h_{-4} := \tfrac{18\sqrt{2}}{512},\ h_{-3} := -\tfrac{\sqrt{2}}{32},\ h_{-2} := -\tfrac{63\sqrt{2}}{512},\ h_{-1} := \tfrac{9\sqrt{2}}{32}, \\ h_0 &:= \tfrac{348\sqrt{2}}{512},\ h_1 := \tfrac{9\sqrt{2}}{32},\ h_2 := -\tfrac{63\sqrt{2}}{512},\ h_3 := -\tfrac{\sqrt{2}}{32},\ h_4 := \tfrac{18\sqrt{2}}{512},\ h_5 := 0,\ h_6 := -\tfrac{\sqrt{2}}{512} \\ g_{-4} &:= \tfrac{\sqrt{2}}{32},\ g_{-3} := 0,\ g_{-2} := -\tfrac{9\sqrt{2}}{32},\ g_{-1} := \tfrac{16\sqrt{2}}{32}, g_0 := -\tfrac{9\sqrt{2}}{32},\ g_1 := 0,\ g_2 := \tfrac{\sqrt{2}}{32}, \end{aligned}$$

we obtain the constants α_L in Table 3 by numerical evaluation. For downward and alternating normalizations, the constants α_L are exact for $L \le 3$. For downward normalization with $a = 1$ and $a' = 1$ the lifting scheme is analogous to [4], formula (4.5). For alternating normalization we find with $a = 1$, $a' = \frac{1}{2}$ in the even decomposition step,

$$\begin{aligned} d_k^{2j} &:= \tfrac{1}{2}\, s^{2j-1}_{(2k-1)\mathrm{mod}\ n_{2j-1}} + \tfrac{1}{32}\ \Big((s^{2j-1}_{(2k+2)\mathrm{mod}\ n_{2j-1}} + s^{2j-1}_{(2k-4)\mathrm{mod}\ n_{2j-1}}) \\ &\qquad -9(s^{2j-1}_{2k\mathrm{mod}\ n_{2j-1}} + s^{2j-1}_{(2k-2)\mathrm{mod}\ n_{2j-1}})\Big)\,, \\ s_k^{2j} &:= s^{2j-1}_{2k\mathrm{mod}\ n_{2j-1}} + \tfrac{1}{16}\ \Big(9(d^{2j}_{k\mathrm{mod}\ n_{2j}} + d^{2j}_{(k+1)\mathrm{mod}\ n_{2j}}) - (d^{2j}_{(k+2)\mathrm{mod}\ n_{2j}} + d^{2j}_{(k-1)\mathrm{mod}\ n_{2j}})\Big)\,, \end{aligned}$$

both for $k = 0, \ldots, n_{2j} - 1$. In the odd decomposition step, again a multiplication with 2 is necessary.

level L	normalized case	alternating case	downward case	n
	α_L	α_L	α_L	
1	2.2980970	1.625	2.25	$n \ge 14$
2	2.6446823	2.25	2.875	$n \ge 32$
3	3.0986707	2.3741493225	3.74658966064453125	$n \ge 64$
4	3.3465789	2.7511579990	4.49393892288208008	$n \ge 128$
5	3.5479124	2.8032060782	5.32267948985099792	$n \ge 256$
6	3.6810877	3.0006566071	6.11064214445650578	$n \ge 512$
7	3.7785195	3.0242370584	6.91897304530721158	$n \ge 1024$
8	3.8462564	3.1240677539	7.71711759639583761	$n \ge 2048$
9	3.8945669	3.1355537412	8.52035180566917916	$n \ge 4096$
10	3.9285788	3.1856066737	9.32104323659601164	$n \ge 8192$

Table 3. Constants α_L for the biorthogonal $(4, 4)$ interpolatory transform.

Biorthogonal $(2+2, 2)$ transform (see [4], formula (4.6)). For the normalized analysis filter coefficients

$$\begin{aligned} h_{-2} &:= -\tfrac{\sqrt{2}}{8},\ h_{-1} := \tfrac{\sqrt{2}}{4},\ h_0 := \tfrac{3\sqrt{2}}{4},\ h_1 := \tfrac{\sqrt{2}}{4},\ h_2 := -\tfrac{\sqrt{2}}{8}, \\ g_{-6} &:= -\tfrac{\sqrt{2}}{256},\ g_{-5} := \tfrac{\sqrt{2}}{128},\ g_{-4} := \tfrac{7\sqrt{2}}{256},\ g_{-3} := 0,\ g_{-2} := -\tfrac{35\sqrt{2}}{128},\ g_{-1} := \tfrac{62\sqrt{2}}{128}, \\ g_0 &:= -\tfrac{35\sqrt{2}}{128},\ g_1 := 0, g_2 := \tfrac{7\sqrt{2}}{256},\ g_3 := \tfrac{\sqrt{2}}{128},\ g_4 := -\tfrac{\sqrt{2}}{256} \end{aligned}$$

we give the constants α_L in Table 4. For downward normalization with $a = 1$ and $a' = 1$ the lifting scheme is analogous to [4], formula (4.6), where $\alpha = \beta = \frac{1}{8}$ and $\gamma = 0$. For alternating normalization we find in the even decomposition step:
Compute all for $k = 0, \ldots, n_{2j} - 1$,

$$\begin{aligned} d_k^{(1)} &:= \tfrac{1}{2}\, s^{2j-1}_{(2k+1)\mathrm{mod}\ n_{2j-1}} - \tfrac{1}{4}\,(s^{2j-1}_{(2k+2)\mathrm{mod}\ n_{2j-1}} + s^{2j-1}_{2k\mathrm{mod}\ n_{2j-1}}), \\ s_k^{2j} &:= s^{2j-1}_{2k\mathrm{mod}\ n_{2j-1}} + \tfrac{1}{2}\,(d^{(1)}_{k\mathrm{mod}\ n_{2j}} + d^{(1)}_{(k-1)\mathrm{mod}\ n_{2j}}), \\ d_k^{2j} &:= d^{(1)}_{(k-1)\mathrm{mod}\ n_{2j}} + \tfrac{1}{32}\ \Big(s^{2j}_{(k-2)\mathrm{mod}\ n_{2j}} - s^{2j}_{(k-1)\mathrm{mod}\ n_{2j}} - s^{2j}_{k\mathrm{mod}\ n_{2j}} + s^{2j}_{(k+1)\mathrm{mod}\ n_{2j}}\Big)\,. \end{aligned}$$

level L	normalized case	alternating case	downward case	n
	α_L	α_L	α_L	
1	2.2760000	1.609375	2.21875	$n \geq 12$
2	2.5807785	2.16656494140625	2.7374267578125	$n \geq 20$
3	2.9904653	2.2890557050704956	3.56502103805542	$n \geq 48$
4	3.2135462	2.6327996477484703	4.24502405524254	$n \geq 112$
5	3.3946980	2.6766678149942891	4.9991885156487	$n \geq 224$
6	3.5145021	2.8563973457742691	5.7162510034054	$n \geq 448$
7	3.6021457	2.8761328739465390	6.4518820033799	$n \geq 896$
8	3.6630830	2.9669840192730337	7.1782317553346	$n \geq 1792$
9	3.7065384	2.9765761919642828	7.9092225548303	$n \geq 3584$
10	3.7371365	3.0221252555286941	8.6378929721518	$n \geq 7168$

Table 4. Constants α_L for the biorthogonal $(2+2,2)$ transform.

Daubechies D4-transform (see [10]).
Let the normalized coefficients be given by

$$h_0 := \tfrac{1+\sqrt{3}}{4\sqrt{2}},\ h_1 := \tfrac{3+\sqrt{3}}{4\sqrt{2}},\ h_2 := \tfrac{3-\sqrt{3}}{4\sqrt{2}},\ h_3 := \tfrac{1-\sqrt{3}}{4\sqrt{2}}$$

and $g_{-2} := h_3$, $g_{-1} := -h_2$, $g_0 := h_1, g_1 := -h_0$. By orthonormality, we have that $\tilde{g} = g$ and $\tilde{h} = h$. For the periodic discrete wavelet transform through 1 level we find

$$\alpha_1 = h_0 + h_2 + g_0 + |g_2| = \tfrac{3+\sqrt{3}}{2\sqrt{2}} \approx 1.673032607.$$

One possible algorithm for the first decomposition level based on lifting is then of the form:
Compute

$$\begin{array}{lll}
d_k^{(1)} & := s^0_{(2k+1)\text{mod } n} - \frac{\sqrt{3}}{3}\, s^0_{2k\text{mod } n}, & k = 0, \ldots, \frac{n}{2} - 1, \\
s_k^{(1)} & := s^0_{2k\text{mod } n} + \frac{\sqrt{3}}{2}\, d^{(1)}_{k\text{mod } n/2}, & k = 0, \ldots, \frac{n}{2} - 1, \\
d_k^{(2)} & := d^{(1)}_{k\text{mod } n/2} - \frac{\sqrt{3}}{3}\, s^{(1)}_{k\text{mod } n/2} + (\sqrt{3}-\sqrt{2})(\sqrt{2}-1)\, s^{(1)}_{(k-1)\text{mod } n/2}, & k = 0, \ldots, \frac{n}{2} - 1, \\
s_k^1 & := s^{(1)}_{k\text{mod } n/2} - \frac{\sqrt{3}-1}{2\sqrt{2}}\, d^{(2)}_{(k+1)\text{mod } n/2}, & k = 0, \ldots, \frac{n}{2} - 1, \\
d_k^1 & := d^{(2)}_{k\text{mod } n/2} + (\sqrt{3}-\sqrt{2})(\sqrt{2}-1)\, s^1_{(k-1)\text{mod } n/2}, & k = 0, \ldots, \frac{n}{2} - 1,
\end{array}$$

and put $\mathbf{y}^1 := \text{rd}\, \alpha_1\, (\mathbf{s}^1, \mathbf{d}^1)^T$.
Note that this lifting method produces the normalized wavelet transform through one level for the D4 filter bank. Another method (see [4], Example 4.3), where the analysis filters h and g are multiplied with the factors $\frac{\sqrt{3}+1}{\sqrt{2}}$ and $\frac{\sqrt{3}-1}{\sqrt{2}}$ (i.e., $a = \sqrt{3}+1$, $a' = 1$), is given by:
Compute

$$\begin{array}{lll}
d_k^{(1)} & := s^0_{(2k+1)\text{mod } n} - \sqrt{3}\, s^0_{2k\text{mod } n}, & k = 0, \ldots, \frac{n}{2} - 1, \\
s_k^1 & := s^0_{2k\text{mod } n} + \frac{\sqrt{3}}{4}\, d^{(1)}_{k\text{mod } n/2} + \frac{\sqrt{3}-2}{4}\, d^{(1)}_{(k-1)\text{mod } n/2}, & k = 0, \ldots, \frac{n}{2} - 1, \\
d_k^1 & := d^{(1)}_{k\text{mod } n/2} + s^1_{(k+1)\text{mod } n/2}, & k = 0, \ldots, \frac{n}{2} - 1,
\end{array}$$

and put $\mathbf{y}^1 := \mathrm{rd}\,\alpha\,(\mathbf{s}^1, \mathbf{d}^1)^T$.

Here, one needs to choose $\alpha \geq \alpha_1$ with

$$\alpha_1 := \tfrac{\sqrt{3}-1}{\sqrt{2}}(h_0 + h_2) + \tfrac{\sqrt{3}+1}{\sqrt{2}}\,(|g_0| + |g_{-2}|) = \tfrac{1}{2} + \sqrt{3}.$$

Application of this last algorithm for $L \geq 1$ leads to a special "upward normalization", where, compared with the normalized linear wavelet transform, $\mathbf{d}^j$ is multiplied with $\alpha_L\,(\frac{\sqrt{3}+1}{\sqrt{2}})^{j-1}\,(\frac{\sqrt{3}-1}{\sqrt{2}})$ and $\mathbf{s}^L$ with $\alpha_L\,(\frac{\sqrt{3}+1}{\sqrt{2}})^L$. For $L > 1$ the numerically evaluated constants α_L can be found in Table 5.

level L	normalized case	upward case	n
	α_L	α_L	
1	1.6730326	2.232050807	$n \geq 4$
2	2.1646385	3.293231463	$n \geq 12$
3	2.5178072	3.419382798	$n \geq 24$
4	2.7694977	3.401977285	$n \geq 48$
5	2.9481636	3.381568411	$n \geq 96$
6	3.0747447	3.369532292	$n \geq 192$
7	3.1643379	3.362061609	$n \geq 384$
8	3.2277205	3.357187157	$n \geq 768$
9	3.2725496	3.354487871	$n \geq 1536$
10	3.3042523	3.353066838	$n \geq 3072$

Table 5. Constants α_L for Daubechies $D4$ filter bank in the normalized case and the special "upward normalization" with $a = 1 + \sqrt{3}$ and $a' = \sqrt{3} - 1$ in each level.

References

[1] M.D. Adams, *Reversible integer-to-integer wavelet transforms for image coding*, PhD theses, Department of Electrical and Computer Engineering, The University of British Columbia, 2002.

[2] M.D. Adams and F. Kossentini, Reversible integer-to-integer wavelet transforms for image compression: Performance evaluation and analysis, IEEE Trans. Image Process. **9** (6) (2000), 1010–1024.

[3] C. Brislawn, Classification of nonexpansive symmetric extension transforms for multirate filter banks, Appl. Comput. Harmon. Anal. **3** (1996), 337–357.

[4] A.R. Calderbank, I. Daubechies, W. Sweldens, and B.L. Yeo, Wavelet transforms that map integers to integers, Appl. Comput. Harmon. Anal. **5** (1998), 332 – 369.

[5] W.K. Cham and P.C. Yip, Integer sinusoidal transforms for image processing, Internat. J. Electron. **70** (1991), 1015 – 1030.

[6] H. Chao, P. Fisher, and Z. Hua, An approach to integer wavelet transformations for lossless image compression, Lect. Notes Pure Appl. Math. **202** (1998), 19–38.

[7] Y.-J. Chen, S. Oraintara and T.Q. Nguyen, Integer discrete cosine transform (Int-DCT), preprint, 2000.

[8] L.Z. Cheng, H. Xu and Y. Luo: Integer discrete cosine transform and its fast algorithm, Electronics Letters **37**, 2001, 64–65.

[9] A. Cohen, I. Daubechies and J.-C. Feauveau, Biorthogonal bases of compactly supported wavelets, Commun. Pure Appl. Math. **45** (1992), 485–560.

[10] I. Daubechies, *Ten Lectures on Wavelets*, SIAM, Philadelphia, 1992.

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

[12] S. Dewitte and J. Cornelis, Lossless integer wavelet transform, IEEE Signal Processing Letters, **4** (6) (1997), 158–160.

[13] P. Hao and Q. Shi, Matrix factorizations for reversible integer mapping, IEEE Trans. Signal Process. **49** (10) (2001), 2314–2324.

[14] H.Y. Jung and R. Prost, Lossless subband coding system based on rounding transform, IEEE Trans. on Signal Processing **46** (6) (1998), 2535–2540.

[15] H. Kim and C.C. Li, Lossless and lossy image compression using biorthogonal wavelet transforms with multiplierless operations, IEEE Trans. on Circuits and Systems II **45** (8) (1998), 1113–1118.

[16] K. Komatsu and K. Sezaki, Reversible discrete cosine transform, Proc. IEEE ICASSP98, 1998, 1769-1772.

[17] X. Li, B. Tao, and M.T. Orchard, On implementing transforms from integers to integers, In Proc. of IEEE International Conference on Image processing, vol. 3, Chicago, IL, USA, 1998, 881–885.

[18] J. Liang and T.D. Tran, Fast multiplierless approximations of the DCT: the lifting scheme, IEEE Trans. Signal Process. **49** (2001), 3032 – 3044.

[19] C. Loeffler, A. Lightenberg, and G. Moschytz, Practical fast 1–d DCT algorithms with 11 multiplications, Proc. IEEE Internat. Conf. Acoust. Speech Signal Process., vol. 2 (1989), 988 – 991.

[20] S. Oraintara, Y.J. Chen, and T.Q. Nguyen, Integer fast Fourier transform, IEEE Trans. on Signal Processing **50**(3) (2002), 607–618.

[21] W. Philips, Lossless DCT for combined lossy/lossless image coding, Proc. IEEE Internat. Conf. Image Process., vol. 3, 1998, pp. 871 – 875.

[22] G. Plonka and M. Tasche, Split–radix algorithms for discrete trigonometric transforms, Preprint, Gerhard–Mercator–Univ. Duisburg, 2002.

[23] G. Plonka and M. Tasche: Reversible integer DCT algorithms, Preprint, Gerhard–Mercator–Univ. Duisburg, 2002.

[24] M. Primbs, Integer DCT Algorithmen, Diplomarbeit, Institute of Mathematics, Gerhard-Mercator Universität Duisburg, 2003.

[25] M. Rabbani, and P.W. Jones, *Digital Image Compression Techniques*, Bellingham, WA:SPIE, 1991.

[26] K.R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*, Academic Press, Boston, 1990.

[27] A. Said and W.A. Pearlman, An image multiresolution representation for lossless and lossy compression, IEEE Trans. on Image Processing **5** (9) (1996), 1303–1310.

[28] G. Strang and T. Nguyen, *Wavelets and Filter Banks*, Wellesley-Cambridge Press. Wellesley, MA, USA, 1996.

[29] T.D. Tran, The BinDCT: Fast multiplierless approximation of the DCT, IEEE Signal Process. Lett. **7** (2000), 141–144.

[30] C.F. Van Loan, Computational Framework for the Fast Fourier Transform, SIAM, Philadelphia, 1992.

[31] Y. Zeng, L. Cheng, G. Bi, and A.C. Kot, Integer DCTs and fast algorithms, IEEE Trans. Signal Process. **49** (2001), 2774–2782.