UNIVERSITÄT SALZBURG

Approximating Linear Image Operations in the Wavelet Domain

Rade Kutil

Technical Report 2005-08 September 2005

Department of Scientific Computing

Scientific Computing

Jakob-Haringer-Straße 2
5020 Salzburg
Austria
www.scicomp.sbg.ac.at

Technical Report Series

Approximating Linear Image Operations in the Wavelet Domain

Rade Kutil

Salzburg University, Dept. of Scientific Computing, Austria
rkutil@cosy.sbg.ac.at
http://www.scicomp.sbg.ac.at/staff/rade.kutil.html

Abstract

Wavelet based methods get more and more widely used in image processing applications. This increases the need for standard image operations to be performed in the wavelet domain. As the wavelet transform lacks some basic properties such as shift invariance and a convolution theorem similar to the Fourier transform, this turns out to be a non trivial task. This work presents a strategy to implement linear image operations in the wavelet domain. Experimental results are shown for the pixel-wise product, the shift operation and the convolution. Because of good results for the convolution, this operation is optimised for timing performance to show that it is a competitive alternative to the pixel based operation.

1 Motivation

Wavelet transformed data can nowadays be found in several image coding algorithms [1] and standards [2]. A major disadvantage of the wavelet transform is that there are few operations which can be performed efficiently in the wavelet domain (e.g. denoising). Note that the discrete cosine transform (DCT) owns the same problem. This makes it difficult to manipulate encoded data.

There are several reasons to perform standard image operations in the wavelet domain:

- If a wavelet compressed image has to be modified or analysed, it usually has to be decompressed completely before (and compressed after) the desired operations are performed on non-compressed data. If the operation can be performed in the wavelet domain, the wavelet transform can be avoided in the compression/decompression process. Note that the wavelet transform is usually the most time consuming part of a wavelet compression scheme.

- If it is sufficient to approximate the operation by limiting it to the most important (or coarse scale) wavelet coefficients, execution time can be saved. This means that the operation can be approximated.

- In some cases, there is evidence that the operation is faster when performed on transformed data. The convolution operation, for instance, is

very time consuming. However, the convolution theorem for the Fourier transform shows that a frequency based decomposition can help to speed up the operation.

Apart from intensity based operations on images (which are, of course, easily adapted to wavelet coefficients), the shift operation is the most simple operation on pixels. Unfortunately, the wavelet transform is known as not shift invariant [3]. Therefore, the shift operation will be a complicated task in the wavelet domain. Nevertheless, it is included in this discussion for the sake of completeness.

Another two important operations, which are closely related to each other, are the pixel-wise product and the convolution. Whereas for the Fourier transform, the existence of the convolution theorem shows in an analytical way that the pixel-wise product transforms into convolution when changing from pixels to Fourier coefficients and vice versa, there is no such theorem for the wavelet transform [4]. Only in some special cases, a similar theorem can be stated [5].

The application of the pixel-wise multiplication includes masking of image regions and shading. Furthermore, overwriting (or overlaying) of image regions must be divided into two steps: masking out the covered image regions and adding the desired new image parts. Again, masking involves pixel-wise multiplication. Adding pixels simply corresponds to adding wavelet coefficients in the transformed domain.

The application of the convolution is the search for image objects such as licence plates. Note that the maximum of convolution result yields the minimum mean squared error (MSE) at the corresponding displacement position. Thus, after the convolution of an object (smaller image) with an image, the maximum of the resulting convolution image indicates the position of the best match between the object and the image. The high complexity of the convolution (n^2) longs for a speedup because it is usually too costly. Apart from that, there are many 1-D applications using convolution in signal processing (e.g. filtering with large filters).

2 A Universal Approach

One can consider an image as the sum of a collection of wavelet functions (wavelets). The wavelets' magnitude is determined by their wavelet coefficients. If an operation is performed on the image, the operation garbles the wavelet functions. Therefore, one wavelet falls into several other wavelets. This means that one wavelet coefficient affects several neighbouring coefficients when an operation is performed. The problem is now to find a direct way to calculate these energy relocations.

2.1 Transformation of the Linear Operator

The basic idea is that – as the wavelet transform is a linear operation on image data (representable by a matrix W) – each linear operation on image data can be transformed into a linear operation on transformed data. Let x be a vector that contains all image pixels and W the matrix that performs the wavelet transform such that the transformed vector $\hat{x} = Wx$ contains all wavelet coefficients. Let furthermore O and b be a matrix and a vector representing a linear image operation such that $y = Ox + b$ is the resulting image after application of the

linear operation. Because image addition simply transforms into addition of of wavelet coefficients, we will, henceforth, ignore b by setting it 0.

Now, the transformed operator $\hat{O}$ that performs the same operation as O in the wavelet domain ensues as follows:

$$\hat{y} = Wy = WOx = WOW^{-1}\hat{x} =: \hat{O}\hat{x}$$

Note that all of the operations mentioned above are linear operations: For the shift operation, O basically consists of shifted identity matrices. For the product operation, O is a diagonal matrix with the pixels of the multiplier image as diagonal elements. For the convolution operation, the rows of O are vectors containing shifts of the object image (the image that is to be convolved with the original image). This means that the convolution image is hidden in the operator O.

2.2 Calculation of Operator Elements

To calculate the elements of $\hat{O}$, we choose a very simple approach following the above equation:

$$\hat{O}_{i,j} = (WOW^{-1}e_j)_i$$

Here, e_j is a unit vector with all elements set to 0 except for the j-th coefficient which is set to 1. Thus, we calculate the wavelet $W^{-1}e_j$ corresponding to coefficient j, perform the operation O on it and transform the result back into the wavelet domain again. There we can read off the columns of the matrix $\hat{O}$.

As the matrices O and $\hat{O}$ have a size of the square of the image size, it is not possible to store all matrix elements in memory. However, depending on the operation, many elements will be 0 or near 0. Only a relatively small amount of elements is kept in these lists. If too many elements are non-zero, small elements are discarded – resulting in an approximation of the operation.

So, we hope that only a small subset of the elements is significant and the rest can be neglected. The significant elements $\hat{O}_{i,j}$ shall be stored in a list together with their positions (i, j).

Of course, this process is very time consuming. Therefore, applications are preferable in which the calculation of $\hat{O}$ can be done once "off-line" and $\hat{O}$ is applied several times. For instance, think of looking for licence plates in the frames of a video sequence. $\hat{O}$ only depends on the shape of the licence plate. It can be calculated once and used for convolution in each frame of the video sequence.

2.3 Significance Weighting

The decision which element $\hat{O}_{i,j}$ to drop and which to store is based on its significance $s_{i,j}$ which is basically its absolute value: $s_{i,j} = |\hat{O}_{i,j}|$. However, as the coefficients of lower frequency sub-bands (higher decomposition level) usually are bigger, the significance of corresponding matrix elements should be higher. We choose to multiply the significance by $2^{l(n)}$ where $l(n)$ is the decomposition level of the coefficient $\hat{x}_n$. As there are two coefficients associated with each single matrix element ($\hat{y}_i$ and $\hat{x}_j$ for $\hat{O}_{i,j}$), there are four possibilities: *source* – $s_{i,j} = |\hat{O}_{i,j}|2^{l(i)}$, *dest* – $s_{i,j} = |\hat{O}_{i,j}|2^{l(j)}$, *both* – $s_{i,j} = |\hat{O}_{i,j}|2^{l(i)+l(j)}$ and *bare* – $s_{i,j} = |\hat{O}_{i,j}|$.

3 Implementation Issues

3.1 Matrix Redundancies

For distinct operations, $\hat{O}$ has distinct properties:

- $\hat{O}$ is symmetric if and only if O is symmetric which is true for the product and the convolution operation.
- If the operation is shift invariant (i.e. $OS_d x = S_d O x$ for all d and x, where S_d shifts image x by 2-D vector d), then $\hat{O}$ satisfies the condition $\hat{O}_{i,j} = \hat{O}_{i+d,j+d}$ where $i+d$ is the index of the coefficient which is located in the same sub-band as x_i but translated within this sub-band by $d \cdot 2^{-l(i)}$. Of course, the coordinates of d must be divisible by $2^{\max(l(i),l(j))}$. This restricted shift invariance is valid for the shift and the convolution operation.

These two properties can be used to reduce the memory demands for the storage of $\hat{O}$ as well as the computation time for calculating the elements of $\hat{O}$.

3.2 Performance of Operator Generation

Although the calculation of $\hat{O}$ is not considered as the time-critical part of the method, it owns such a high complexity that we will take a look at how to optimise it.

The calculation of the wavelets $W^{-1}e_j$ can be reduced due to the following facts:

- A whole wavelet transform is not really needed. Each reverse filtering step only needs to process coefficients created in the previous step.
- A 2-D wavelet is an outer product of two 1-D wavelets.
- Only one wavelet $W^{-1}e_0$ has to be calculated for each sub-band. The other wavelets are shifts of this wavelet: $W^{-1}e_{0+d} = S_d(W^{-1}e_0)$.

The application of O on the wavelet $W^{-1}e_j$ can be reduced to those pixels that are located within the wavelet. The shape of the result $OW^{-1}e_j$ depends on the operation. The shift and product operations do not extend the size of $W^{-1}e_j$; the convolution increases the size by the object image size horizontally and vertically. Thus, the subsequent wavelet transform can be reduced to the size of $OW^{-1}e_j$.

The insertion of elements in the storage object can be a critical part of the process if the size of the storage is big. Therefore, it has to be implemented carefully. We choose to use the *set*-class of the C++ standard template library (STL) which is a sorted balanced tree (to be precise: a red-black tree). Thus, element insertion as well as the removal of the smallest element has logarithmic complexity.

4 Experimental Results

For experimentation we will use a square image with 256×256 pixels and 8 bpp. It will be decomposed to depth 5 using the 8-tap orthogonal Daubechies

filter with cyclic border handling. Note that the decomposition (as well as the reconstruction) involves $2 \cdot 8 \cdot (256^2 + 128^2 + 64^2 + 32^2 + 16^2) = 1,396,736$ multiplications and if data is given in wavelet coefficients the alternative to each operation performed in the wavelet domain is to perform a reconstruction and apply the operation as usual on pixels followed by a wavelet decomposition if necessary. The number of multiplications necessary to obtain a reasonable image quality when performing the operation in the wavelet domain will determine if and in which cases our method is useful.

4.1 Shift Operation

Figure 1: Shifted image by pixel (left) and in wavelet domain after 10000 multiplications (right)

Figure 1 shows the result of a sample shift operation in the wavelet domain and pixel-wise for comparison. Because shifts by a power of 2 are easier (high frequency sub-bands can simply be shifted), odd numbers are chosen as coordinates of the shift vector: 87 horizontally and 193 vertically. Here and in further examples, a small number of multiplications (10000, i.e. 0.15 multiplications per pixel) is chosen to make the errors visible.

Figure 2 shows the quality of the shift operation in the wavelet domain depending on the number of multiplications (i.e. the number of matrix elements kept in the storage). Compared to the alternative of wavelet reconstruction, pixel-wise shift plus reconstruction, the break-even point (same number of multiplications) is reached at a PSNR of about 35 dB.

One can see that the methods *source* and *both* for weighting the significance of elements of $\hat{O}$ produces the best results. This means that it is important to process coefficients with bigger expected values first. This result also holds for all other operations.

4.2 Product Operation

Results of the product operation can be seen in Figure 3. A circular mask is chosen as image for the pixel-wise product. The relevant errors are located at

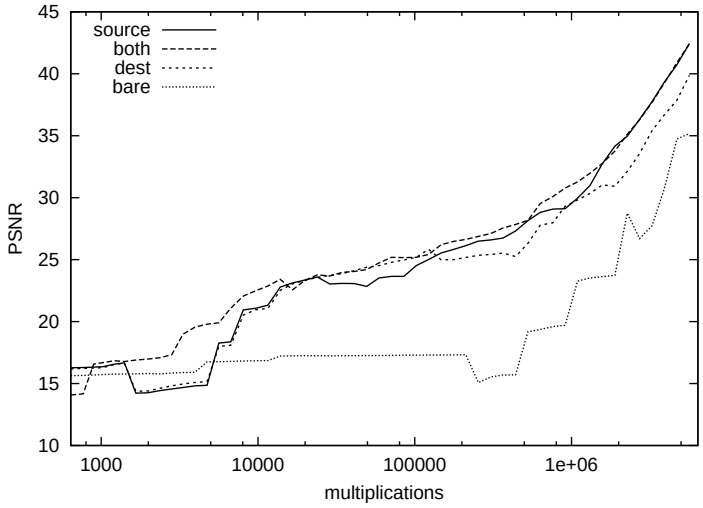

Figure 2: Quality of shift operation

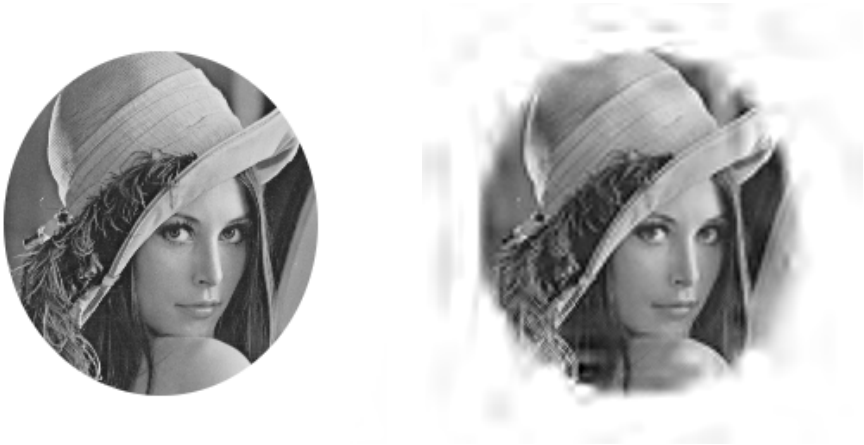

Figure 3: Masked image by pixel (left) and in wavelet domain after 10000 multiplications (right)

the border of the mask. This is where neighbouring coefficients affect each other in a non-trivial way. All wavelet coefficients corresponding to wavelets that are located entirely within the circle simply keep their value. The number of these

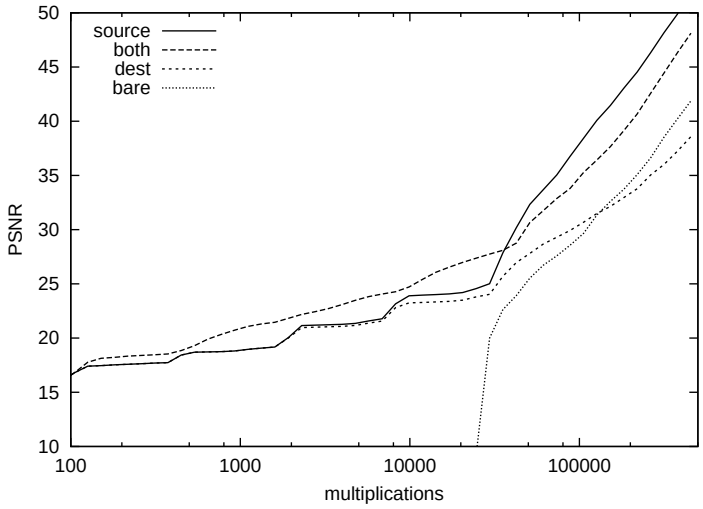

Figure 4: Quality of product operation

coefficients causes the sudden ascent in the curves in Figure 4 above about 20000 multiplications: After they are processed (copied to the destination coefficient set) the complicated handling of mask borders completes the task. This has another consequence: If the mask mainly consists of 1s and 0s (as in this case) then the diagonal elements of $\hat{O}$ can be handled differently. The diagonal of $\hat{O}$ can be stored in a separate array and groups of 1s and 0s can be accumulated, thus, saving a lot of memory.

Compared to the pixel-wise product (involving 256^2 multiplications) not including the wavelet transform, the break-even point (same number of multiplications) is reached at about 40 dB.

4.3 Convolution Operation

For the convolution operation, we choose a part from the center of the image (face) as object image. The convolution should "find" this object image. Figure 5 shows the results. The convolution image is scaled and shifted in terms of grey values so that it covers the whole range of grey values.

Note that the pixel-wise convolution involves more than $4 \cdot 10^9$ multiplications. Figure 6 shows that a very good image quality can be obtained with much less work if the convolution is performed in the wavelet domain. This method can even speed up the convolution if the image and the convolution result must be non-transformed because this adds only $2.8 \cdot 10^6$ multiplications.

4.4 Timing of Convolution

Since the results for the convolution are surprisingly good, we want to take a look at the actual program performance. Both versions of the convolution (direct and

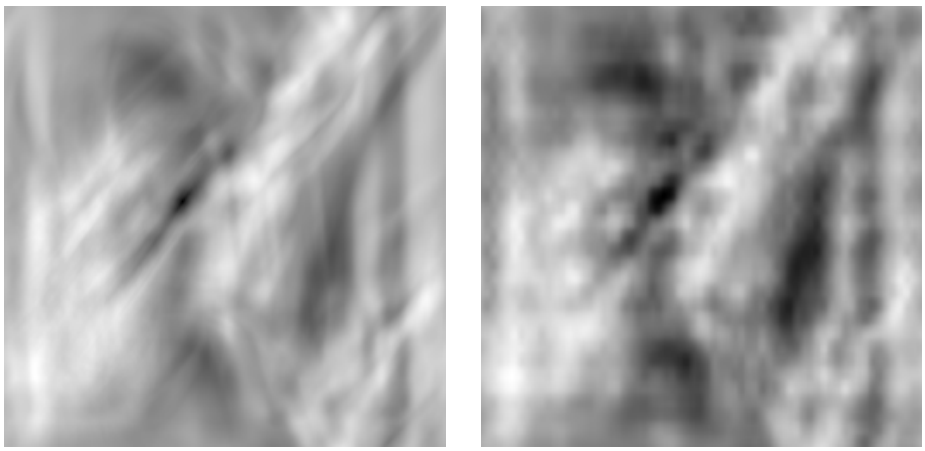

Figure 5: Convolution image by pixel (left) and in wavelet domain after 20000 multiplications (right) for a $60 \times 40 = 2400$ pixel object image. Darker regions are better matches

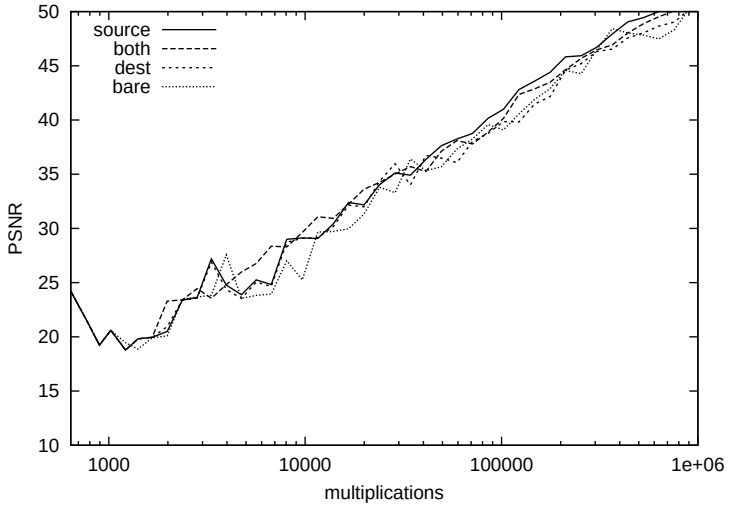

Figure 6: Quality of convolution operation for a $80 \times 60 = 4800$ pixel object image

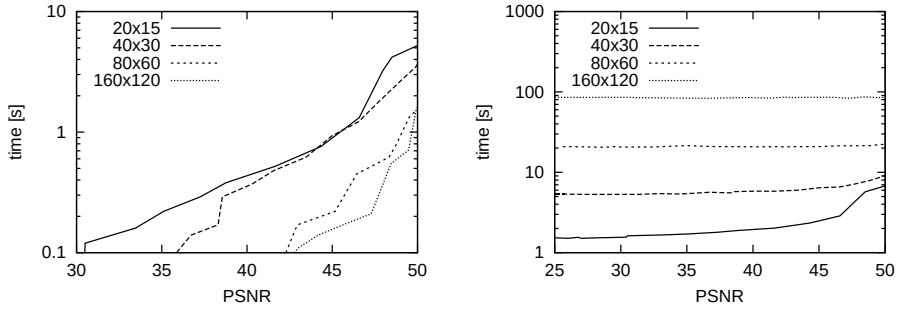

Figure 7: Timing of the convolution without (left) and with operator generation (right). Wavelet decomposition depth is 4

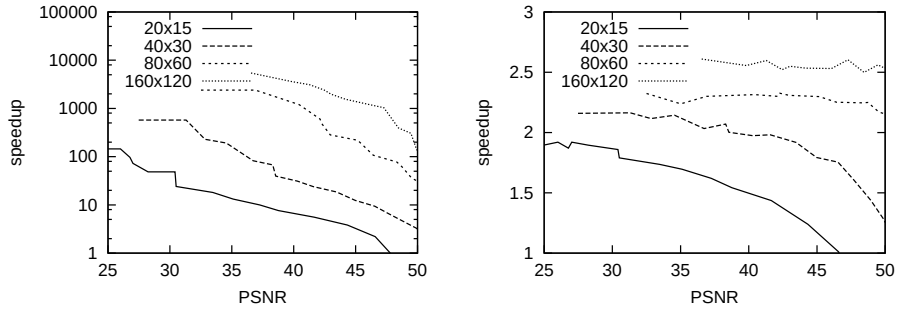

Figure 8: Speedups of the convolution without (left) and with operator generation (right) compared to direct pixel-based convolution

in the wavelet domain) are implemented using double precision floating point numbers. Experiments were conducted on a 400MHz Pentium II.

Figure 7 shows the timing of the convolution in the wavelet domain. Execution times of operator generation (calculation of $\hat{O}$) are separated from those of the application of the operator. Of course, the execution time increases with the desired convolution image quality. The operator generation time also increases with the object image size because operator generation includes the convolution of wavelets and the object image. Interestingly, the execution time of operator application decreases with increasing object image size. The reason is probably that matches for bigger objects can be found more easily. Therefore, the PSNR grows faster.

These results have to be compared to the corresponding execution times for direct pixel-based convolution. This is done in Figure 8. The speedups of operator application are extraordinary. These speedups are only valid if operator generation is "off-line", though. However, even together with operator generation there are noticeable speedups, especially for big object images.

In all timing experiments above, a wavelet decomposition depth of 4 was used. However, the decomposition depth has impact on operator generation time as well as on operator PSNR performance as can be seen in Figure 9. The execution time of operator generation increases with the decomposition depth because the size of $W^{-1}e_j$ increases with the decomposition depth. The ex-

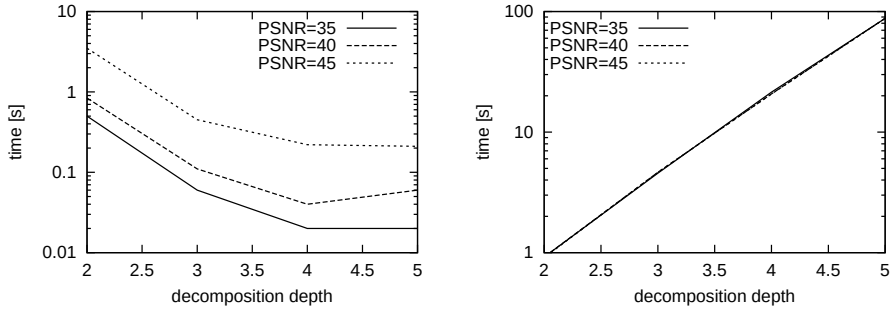

Figure 9: Execution time of convolution operator application (left) and operator generation (right) for varying wavelet decomposition level. Search object size is 80×60

ecution time of operator application decreases with the decomposition depth because the wavelet based convolution simply works better for higher decomposition depths.

5 Conclusions

We have seen that it is not impossible to perform basic image operations in the wavelet domain. A direct method using a list of factors for manipulation of wavelet coefficients is able to transform all linear operations into the wavelet domain (operator generation). Unfortunately, the operator generation may take a lot of computation time and is not or hardly parameterisable. However, if the operation has to be executed several times, the method is able to outperform a direct implementation, especially if image data is given as wavelet coefficients (e.g. in a wavelet compressed format). The method is quality scalable, i.e. less quality can be done faster.

In the case of the convolution operation, speedups are very good. Even together with operator generation, speedups of up to 2.5 are possible.

Acknowledgements

The author was partly supported by the Austrian Science Fund FWF, project no. P13903.

References

[1] Said, A. and Pearlman, W. A. 'A new, fast, and efficient image codec based on set partitioning in hierarchical trees'. IEEE Transactions on Circuits and Systems for Video Technology, June 1996. 6 (3) pp. 243–249

[2] ISO/IEC JPEG committee. 'JPEG 2000 image coding system — ISO/IEC 15444-1:2000', December 2000

[3] Kingsbury, N. G. 'Complex wavelets for shift invariant analysis and filtering of signals'. Applied and Computational Harmonic Analysis, May 2001. 10 (3) pp. 234–253

[4] Lindsey, A. 'The non-existence of a wavelet function admitting a wavelet transform convolution theorem of the fourier type'. Technical report, Ohio University, Athens, Ohio, 1994

[5] Guo, H. and Burrus, C. 'Convolution using the undecimated discrete wavelet transform'. In Proceedings of the Int. Conf. Accust., Speech, Signal Processing, ICASSP-96 (Atlanta, 1996)