Chapter 2

Wavelet transforms on images

Until now we have discussed one dimensional wavelet transforms. Images are obviously two dimensional data. To transform images we can use two dimensional wavelets or apply the one dimensional transform to the rows and columns of the image successively as separable two dimensional transform. In most of the applications, where wavelets are used for image processing and compression, the latter choice is taken, because of the low computational complexity of separable transforms.

Before explaining wavelet transforms on images in more detail, we have to introduce some notations. We consider an $N \times N$ image as two dimensional pixel array $\mathcal{I}$ with N rows and N columns. We assume without loss of generality that the equation $N = 2^r$ holds for some positive integer r.

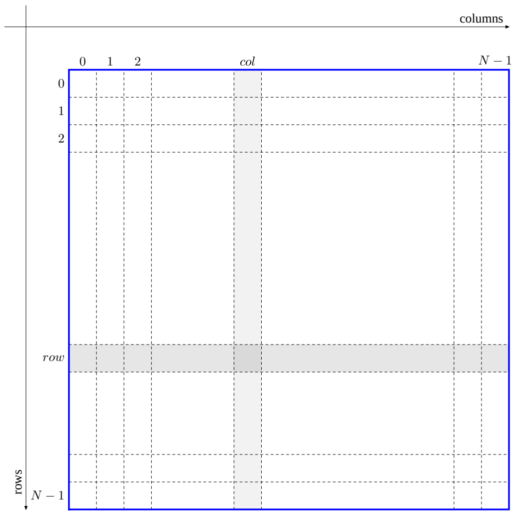


Figure 2.1: images interpretation as two dimensional array $\mathcal{I}$, where the rows are enumerated from top to bottom and the columns from left to right, starting at index zero

In Figure 2.1 we illustrate, how the pixels of the images are arranged in the corresponding array $\mathcal{I}$. The rows are enumerated from top to bottom and the columns from left to right. The index starts with zero and therefore the largest index is $N - 1$. The image pixels themself at row i and column j will be denoted by $\mathcal{I}[i, j]$ or $\mathcal{I}_{i,j}$. The wavelet transformed image will be denoted by $\hat{\mathcal{I}}$ and the coefficients are addressed with $\hat{\mathcal{I}}[k, l]$ or $\hat{\mathcal{I}}_{k,l}$. For the reconstructed image we will use $\tilde{\mathcal{I}}$ and address the corresponding reconstructed pixels as $\tilde{\mathcal{I}}[n, m]$ or $\tilde{\mathcal{I}}_{n,m}$.

The pixels and coefficients themselves are stored as signed integers in two's complementary encoding.

Therefore the range is given as

$$\mathcal{I}[\text{row}, \text{col}] \in \left[-2^{\text{dpth}-1}, 2^{\text{dpth}-1} - 1\right],$$

where $0 \leq \text{row}, \text{col} < N$, assuming a dpth-bit greyscale resolution. Thus, we can distinguish between 2^{dpth} different values of brightness. For an illustration see Figure 2.2. The smallest value $-2^{\text{dpth}-1}$ and

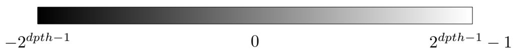


Figure 2.2: greyscales and the corresponding pixel values for dpth-bit resolution

the largest value $2^{\text{dpth}-1} - 1$ correspond to black and white, respectively. As a consequence pixels with magnitude around zero appear as grey color.

In color images each pixel is represented by several color components. Typically there are three of them per pixel. In the RGB color space, e.g., there is one component for red, green, and blue, respectively. Other choices are the YUV color space (luminance and chrominance) and the CMYK color space (cyan, magenta, yellow, black). Note, that there exist YUV based image and video formats, where the size N of the different components is different (e.g. 4:2:2 and 4:1:1). In the case of the 4:1:1 format for instance, we obtain three pixel arrays of size N, $\frac{N}{4}$, and $\frac{N}{4}$.
Throughout this thesis we will treat each color component of color images as separate greyscale image.

Now, let us come back to wavelet transforms on images. As already mentioned the one dimensional transform will be applied to rows and columns successively. Consider a row $r = (r_0, \ldots, r_{N-1})$ of an image $\mathcal{I}$. This row has finite length in contrast to the signals or sequences we have considered until now. In order to convolve such a row r with a filter f we have to extend it to infinity in both directions. Let r' be the extended row defined by

$$r' = (\ldots, r'_0, \ldots, r'_{N-1}, \ldots),$$

where $r'_k = r_k$ for all $0 \leq k < N - 1$. But, how do we have to set the values of r' at positions k with $k \notin [0, N-1]$? In some sense we are free to choose these remaining samples. In the next section we will explain, why reflection at the image boundary should be used in horizontal and in vertical direction.

2.1 Reflection at Image Boundary

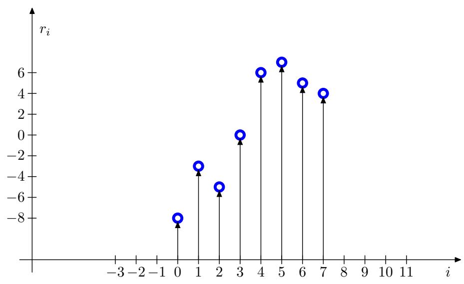
There are several choices to choose the values of r'_k from outside the interval $[0, N-1]$. The most popular one's are

- padding with zeros,
- periodic extension, or
- symmetric extension.

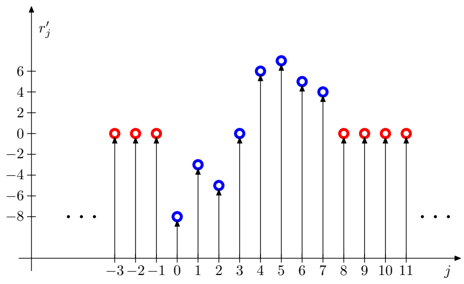
The simplest choice is to set all remaining r'_k to zero. For an illustration see Figure 2.3. In Figure 2.3(a) a sequence r of length $N = 8$ is shown, where $r = (-\mathbf{8}, -3, -5, 0, 6, 7, 5, 4)$. In Figure 2.3(b) this sequence is padded with zeros in order to obtain the infinite sequence r':

$$r'_k = \begin{cases} r_k & : \quad \text{for all } 0 \leq k < N \\ 0 & : \quad \text{else}. \end{cases}$$

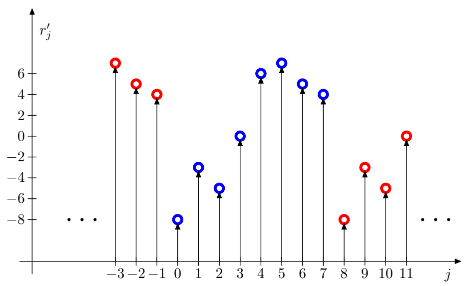
We can observe, that in general there will be discontinuities at the boundary.
The substantial difference between the value of the border coefficients and zero leads to coefficients of large amount in the high frequency subbands. These differences decrease the compression efficiency and introduce artefacts at the boundaries since the reconstructed pixel values depend on the values of the coefficients from outside, too, if lossy compression is considered.



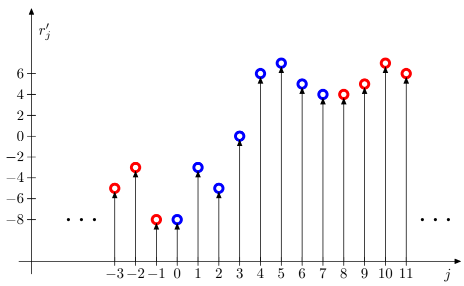
(a) the original sequence r(e.g. image row)



(b) r' obtained from r by zero padding



(c) r' obtained from r by periodic extension



(d) r' obtained from r by symmetric extension

Figure 2.3: different choices for the boundary treatment of finite length sequences, here $r = (-\mathbf{8}, -3, -5, 0, 6, 7, 5, 4)$ of length $N = 8$

The computation of the coefficients $c_{0,0}, c_{0,1}$ and $d_{0,N-1}$ at the first level of a wavelet transform using the CDF(2,2) wavelet depends on pixels from outside. In general, for a symmetric wavelet with corresponding analysis filters $\tilde{h}$ and $\tilde{g}$ the computation of the coefficients

$$\begin{array}{rcl} c_{0,0}, c_{0,1}, \ldots, c_{0,l} & \text{with} & l = \dfrac{\left|\tilde{h}\right| + 1}{2} \\ d_{0,N-1-l}, d_{0,N-l}, \ldots, d_{0,N-1} & \text{with} & l = \dfrac{|\tilde{g}| + 1}{2} \end{array}$$

depends on pixels from outside.

A different approach known from Fourier techniques is periodic extension of the signal, that is

$$r'_{k \cdot N + l} = r_l$$

where $0 \leq l < N$ and $k \in \mathbb{Z}$. Here we encounter the same drawbacks as in the case of padding with zeros (cf. Figure 2.3(c)). The introduced differences at the boundary are considerable as well. Another drawback arises in hardware implementations. Here we have to buffer the first samples of r in order to perform the operations at the end of the sequence r. This can result in large buffers.

The most preferred method for the choice of the coefficients from outside of r is based on symmetric extension. Figure 2.3(d) depicts this method applied to our example sequence. More formally, symmetric extension r' is defined by

$$r'_{k \cdot N + l} = \begin{cases} r_{N-1-l} & : \quad \text{if } k \text{ is an odd value} \\ r_l & : \quad \text{if } k \text{ is an even value.} \end{cases}$$

where $0 \leq l < N$ and $k \in \mathbb{Z}$. The already mentioned difference between coefficients at the boundary do not appear using this kind of extension. Furthermore due to the locally known coefficients no significant, additional amount of buffer memory is required.

In the following we assume, that the boundary treatment was done using symmetric extension. Therefore we do not distinguish between finite and infinite sequence anymore.

2.2 2D-DWT

Now we are able to discuss the separable two dimensional wavelet transform in detail. Consider again a row r of a given image of size $N \times N$. Recall from Section 1.8 the computation of a specific wavelet transform using the Lifting Scheme. After one level of transform we obtain $\frac{N}{2}$ coefficients $c_{0,l}$ and $\frac{N}{2}$ coefficients $d_{0,k}$ with $0 \leq l, k < \frac{N}{2}$. These are given in interleaved order, that is

$$(c_{0,0}, d_{0,0}, c_{0,1}, d_{0,1}, \ldots, c_{0,N-1}, d_{0,N-1}), \tag{2.1}$$

because of the split in odd and even indexed positions in the Lifting Scheme. Usually the row of (2.1) is rearranged to

$$r^{(0)} = (c_{0,0}, c_{0,1}, \ldots, c_{0,N-1}, d_{0,0}, d_{0,1}, \ldots, d_{0,N-1}),$$

because we will apply the transform to the low frequency coefficients $c_{0,l}$ recursively.
Suppose we have already transformed and rearranged all rows of a given image as described above. If we store the computed coefficients in place, that is in the memory space of the original image, we obtain a new array with a structure as shown in Figure 2.4.

(a) *lena*, $N = 512$, $\text{dpth} = 8$

(b) CDF(2,2) wavelet transform applied to the rows of (a)

(c) low (L, $c_{0,k}$) and high (H, $d_{0,k}$) frequency coefficient blocks

Figure 2.4: one dimensional CDF(2,2) wavelet transform applied to the rows and columns of the benchmark image *lena* with reflection at the boundaries

On the left the well known benchmark image *lena*[1] is shown. To the right of it we have applied the CDF(2,2) wavelet transform to the rows of the image. The corresponding result is interpreted as image again (Figure 2.4(b)) and is composed of a coarse and scaled version of the original and the details, which are necessary to reconstruct the image under consideration. On the right we have illustrated this interpretation as low and high frequency coefficients blocks, denoted by L and H, respectively. Remark that most of the

[1]For the curious: 'lena' or 'lenna' is a digitized Playboy centerfold, from November 1972. (Lenna is the spelling in Playboy, Lena is the Swedish spelling of the name.) Lena Soderberg (ne Sjööblom) was last reported living in her native Sweden, happily married with three kids and a job with the state liquor monopoly. In 1988, she was interviewed by some Swedish computer related publication, and she was pleasantly amused by what had happened to her picture. That was the first she knew of the use of that picture in the computer business. (http://www.lenna.org)

high frequency coefficients $d_{0,k}$ are shown in grey color, which corresponds to small values around zero (cf. Figure 2.2).

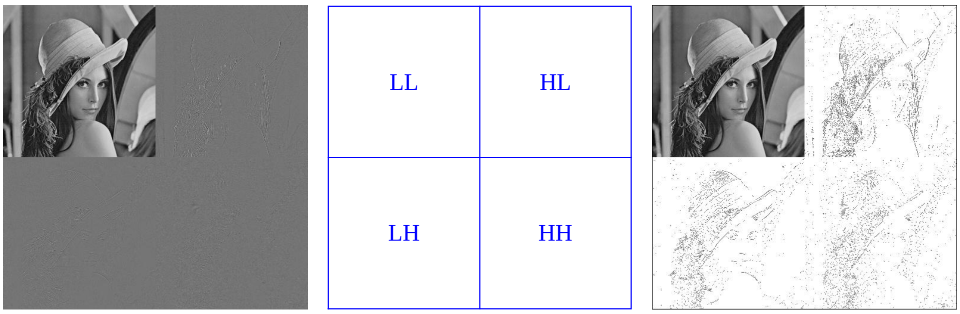
The one dimensional wavelet transform can be applied to the columns of the already horizontal transformed image as well. The result is shown in Figure 2.5 and is decomposed into four quadrants with different interpretations.

LL: The upper left quadrant consists of all coefficients, which were filtered by the analysis low pass filter $\tilde{h}$ along the rows and then filtered along the corresponding columns with the analysis low pass filter $\tilde{h}$ again. This subblock is denoted by LL and represents the approximated version of the original at half the resolution.

HL/LH: The lower left and the upper right blocks were filtered along the rows and columns with $\tilde{h}$ and $\tilde{g}$, alternatively. The LH block contains vertical edges, mostly. In contrast, the HL blocks shows horizontal edges very clearly.

HH: The lower right quadrant was derived analogously to the upper left quadrant but with the use of the analysis high pass filter $\tilde{g}$ which belongs to the given wavelet. We can interpret this block as the area, where we find edges of the original image in diagonal direction.

The two dimensional wavelet transform can be applied to the coarser version at half the resolution, recursively, in order to further decorrelate neighboring pixels of the input image. For an illustration we refer to Figure 2.6. The subbands in the next higher transform levels l will be denoted by $\mathrm{LL}^{(l)}$, $\mathrm{LH}^{(l)}$, $\mathrm{HL}^{(l)}$, and $\mathrm{HH}^{(l)}$, where $\mathrm{LL}^{(1)} = \mathrm{LL}$, $\mathrm{LH}^{(1)} = \mathrm{LH}$, $\mathrm{HL}^{(1)} = \mathrm{HL}$, and $\mathrm{HH}^{(1)} = \mathrm{HH}$, respectively.



(a) CDF(2,2) wavelet applied as tensor product to the rows and columns of the image *lena*

(b) the different frequency blocks

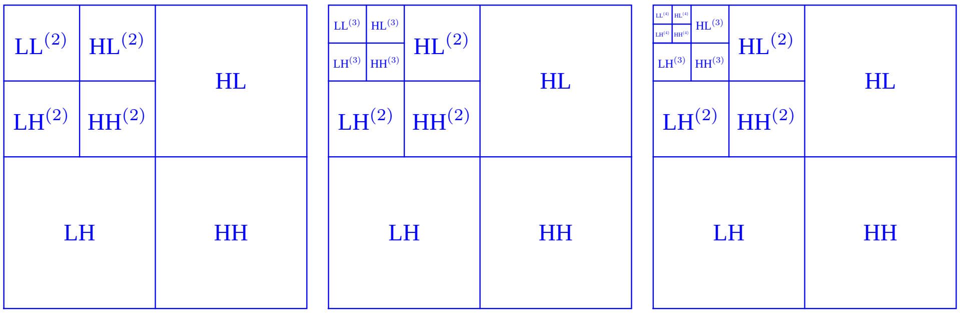
(c) set all coefficients v in LH, HL, HH with $-20 < v < 20$ to white color

Figure 2.5: one dimensional CDF(2,2) wavelet transform applied to the rows of the benchmark image *lena* with reflection at the image boundaries

Since we have restricted the images to be of quadratic size $N = 2^l$ for $l \in N$, we can perform at most $l = \log_2 N$ levels of transform. Thereafter the coefficient in the upper left corner represents the average greyscale value of the whole image and is called *DC coefficient* (DC : direct current). In practice, usually four up to six level of wavelet transform level will be performed.
Due to the local similarities between neighboring pixels, many coefficients in the LH, HL, and HH subbands at different scales will be small. As a consequence only a few samples, especially those of the LL block at the coarsest scale, represents most out of the *images energy*. The *energy* $\mathcal{E}(\mathcal{I})$ of an image $\mathcal{I}$ is defined as

$$\mathcal{E}(\mathcal{I}) = \sum_{r=0}^{N-1} \sum_{c=0}^{N-1} (\mathcal{I}_{r,c})^2 \tag{2.2}$$

This observation is the starting point of wavelet based image compression algorithms, which are explained in Chapter 4.



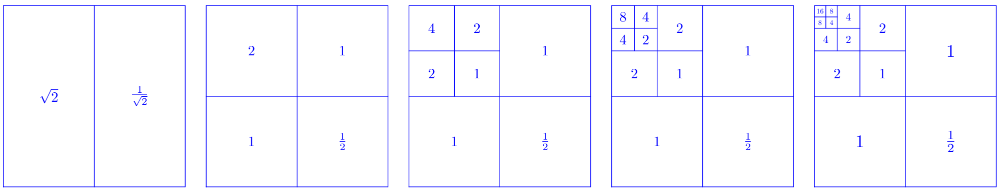
(a) two levels of 2D-DWT (b) three levels of 2D-DWT (c) four levels of 2D-DWT

Figure 2.6: multiresolution scheme after several levels of wavelet transform

In this thesis we will focus on the tree structured decomposition as shown in Figure 2.6, where only the LL blocks are subdivided. This type of image decomposition is also known as *multiresolution scheme* and *multiscale representation*. Other decomposition types are possible and known under the terms of *wavelet packets* [CMQW94].

2.3 Normalization Factors of the CDF(2,2) Wavelet in two Dimensions

In Section 1.7 we have already mentioned the normalization factors of the CDF(2,2)-wavelet for one dimension. To simplify the calculation of the transform we have extracted the factors $\sqrt{2}$ and $\frac{1}{\sqrt{2}}$, which allows us to use efficient integer arithmetic units in hardware implementations.
In order to preserve the average of a one dimensional signal, or the average of the brightness of images, we have to consider the normalization factors after the wavelet transform has taken place.



(a) 1D-DWT (rows) (b) 1D-DWT (columns) (c) 2nd level (d) 3rd level (e) 4th level

Figure 2.7: normalization factors of the CDF(2,2) wavelet in two dimension for each level l, $0 \leq l < 5$

In one dimension we have to scale the low pass coefficients with $\sqrt{2}$ and the high pass coefficients with $\frac{1}{\sqrt{2}}$, which is shown in Figure 2.7(a). Thereafter the same has to be done in vertical direction. Here, the normalization factors become integer powers of two as you can easily verify in Figure 2.7, where each subblock is indexed with the corresponding factor.

To summarize, we can abstract from those normalization factors during the implementation of the CDF(2,2) wavelet transform. Afterwards they will implicit stored and processed. Note, that these implicit factors have no influence to the growth of bitwidth, which is necessary to store the wavelet coefficients.