NTNU
Innovation and Creativity

On the Efficiency of Data Communication for the Ultramonit Corrosion Monitoring System

Tarjei Rommetveit

Master of Science in Communication Technology
Submission date: June 2006
Supervisor: Tor Audun Ramstad, IET
Co-supervisor: Øystein Baltzersen, Sensorlink AS
Arne Solstad, Sensorlink AS

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Problem Description

Ultramonit is a system developed for permanent corrosion monitoring of critical parts of the subsea oil- and gas pipelines based on ultrasound. The ultrasonic signals are digitized immediately at the subsea receiver, but due to a harsh environment, the communication link to the outside world is identified as the system's bottleneck. Hence, real-time compression of the ultrasonic traces is of interest before transmission.

The following problems should be addressed:

-Theoretical description of chosen compression technique(s).
-Evaluation of the usefulness of the chosen compression technique(s) on available electronics (FPGA / MicroBlaze microcontroller.)
-Implementation in C of compression algorithm(s) onto the FPGA/MicroBlaze microcontroller.
-Defining a fidelity measure.
-Optimization of implemented compression algorithm(s) with respect to the defined fidelity measure.

Assignment given: 13. January 2006
Supervisor: Tor Audun Ramstad, IET

Summary

Ultramonit is a system under development for permanent installation on critical parts of the subsea oil- and gas pipelines in order to monitor the corrosion continuously by using ultrasound. The communication link which connects the Ultramonit units with the outside world is identified as the system's bottleneck, and it is thus of interest to compress the ultrasonic data before transmission.

The main goal of this diploma work has been to implement and optimize a lossy compression scheme in C on the available hardware (HW) with respect to a self-defined fidelity measure. Limited resources, such as memory constraints and constraints with respect to the processing time, have been a major issue during implementation. The real-time aspect of the problem results in an intricate relation between transfer time, processing time and compression ratio for a given fidelity.

The encoder is optimized with respect to two different bit allocation schemes, two different filters as well as various parameters. Compared to transferring the unprocessed traces, the results demonstrate that the transfer time can be reduced with a factor 12. This yields acceptable fidelity concerning the main application of long term monitoring of subsea pipelines. However, for ultra-high precision applications where the total change in thickness due to corrosion is less than a few micrometers, compression should not be employed.

Acknowledgements

First of all, I would like to thank my supervisors at SensorLink AS, Arne Solstad and Øystein Baltzersen, for giving good guidance and showing great interest in my work.

Furthermore, I am very thankful to Tor Inge Waag at SensorLink for introducing me to the world of ultra-high precision thickness calculations. I am also grateful to the following people at SesorLink: Arve Skjetne for setting up the corrosion experiments, Arve Fiksdal for never getting tired of my questions related to C-programming and Ibrahim Karage for his ever cheerful attitude.

And finally, I would like to thank my main supervisor, Prof. Tor Audun Ramstad, for his countless advises regarding compression techniques.

Table of contents

Table of figures

List of tables

Abbreviations

µm……micrometer
A/D……Analog to Digital
BWP……Best Wavelet Packet basis in a rate distortion sense
CMF……Conjugate Mirror Filter
DWT……Discrete Wavelet Transform
FPGA……Field Programmable Gate Array
HP……High-Pass
HW……Hardware
IDWT……Inverse Discrete Wavelet Transform
KB……Kilobyte
LP……Low-Pass
MHz……Megahertz
mm……millimeter
MRA……Multiresolution Approximation
mse……mean square error
nm……nanometer
pdf……probability density function
ROV……Remoted Operated Vehicle
SAM……Sensor Array Module
SW……Software
WP……Wavelet Packet

1 Introduction

Ultramonit is a system under development for permanent installation on critical parts of the subsea oil- and gas pipelines in order to monitor the corrosion continuously by using ultrasound. The ultrasonic signals are digitized immediately at the subsea receiver, but due to a harsh environment, the communication link to the outside world, is identified as the system's bottleneck. Underwater operations are expensive, and they involve a significant risk of failure which in general increases with the time of operation. Hence, lowering the transfer time using compression will make the subsea operation less vulnerable and generally make it more attractive to potential customers.

An ultrasonic signal is supposed to be encoded while another one is being sent. Consequently there is a certain relationship between the processing time and the transfer time. In addition, the complexity of the encoder in general influences the compression ratio for a given fidelity. It is further of vital importance that the decoded data has maintained the relevant information, i.e. information related to the thickness of the pipeline. Thus we see that there is an intricate relation between transfer time, processing time, compression ratio and fidelity. The main goal of this diploma work has been to implement a source coding coding scheme on the available HW which addresses the abovementioned issues. By the very fact that no source coding currently is applied, any compression ratio better than 1 will be an improvement.

According to the author's knowledge, the problem addressed herein is relatively unique, both in terms of the data to be compressed and its corresponding fidelity measure, in terms of the unusual environment and in terms of the relationship between transfer- and processing time.

1.1 Outline

The structure of the report is as follows: Chapter 2 gives a brief introduction to the Ultramonit system. Chapter 3 discusses theory related to lossy compression, including source coding fundamentals, scalar quantization and signal decomposition. In chapter 4, signal characteristics of the ultrasonic traces are looked into. Chapter 5 concerns the chosen encoding scheme and associated implementation issues. The results and discussion regarding the performance and optimization of the encoding schemes are presented in chapter 6. Furthermore, the relation between the actual transfer- and processing time as well as the fidelity measure is discussed in this chapter.
At last, conclusions and directions for future work are presented in chapter 7.

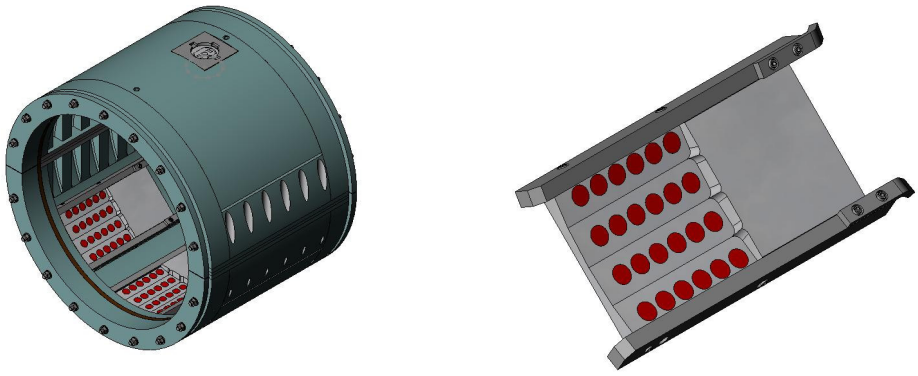
The appendixes deal with the source files for the encoder/decoder, additional theory as well as additional results.

2 The Ultramonit corrosion monitoring system

2.1 An introduction to the system

Ultramonit is a system under development based on the use of *permanently* installed ultrasonic sensors (also referred to as transducers) on subsea pipelines in order to monitor the corrosion continuously. This means that the accuracy/resolution can be improved considerably compared to conventional ultrasonic inspection where the position of the transducer with respect to the pipeline will change each time an inspection occurs. Note that it is the corrosion inside the pipeline that matters; the corrosion on the outer wall is neglectable in comparison.

The instrumentation modules can be installed subsea at very large water depths on existing pipelines. This means that mechanical protection, packaging, and the installation concept must be developed to allow for cost-effective and reliable installation by a Remoted Operated Vehicle (ROV) or a dedicated running tool. The concept of Ultramonit is illustrated in Figure 1, and is based on the use of several Sensor Array Modules (SAMs) which are mounted on the inside of a protective mechanical clamp. Each SAM is manufactured by a piece of plastic which attaches the ultrasound-sensors as well as giving them protection and the correct orientation with respect to the pipeline. Each of these modules is designed for operation with exposure to the target depth hydrostatic pressure.



a) b)

Figure 1 The protective mechanical clamp is shown in a) while one of many SAMs is shown in b).

Connection between the instrumentation modules and the outside world, as well as power supply, is provided by a specially designed inductive coupler which connects directly to the instrumentation modules.

The installation concept assumes that the clamp will be lowered down to the pipeline, and installed. Seals will be closed or activated. The seawater may, if necessary, be displaced by suitable oil to keep the electronics etc. inside the clamp from corroding. For interrogation of the sensors, a ROV may be used (see Figure 2), or the instrumentation may be permanently wired up to existing infrastructure (for example a subsea template).

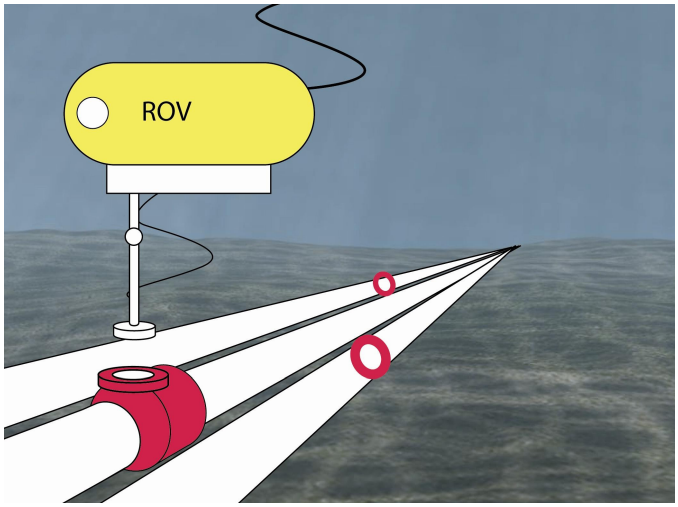


Figure 2 Supply of power and data acquisition by a ROV

2.2 Relevant HW specifications

The SAM is realized with a Xilinx Spartan3 FPGA [1] (Field Programmable Gate Array). A soft-core [2] 50 MHz microcontroller unit denoted MicroBlaze™ [3] is implemented into the FPGA. It is possible to program the MicroBlace with standard C by using the Xilinx Embedded Development Kit/ Xilinx Platform Studio. To support signal processing, 1 MB RAM is available on each SAM in addition to 32 KB initial block-RAM.

The analog to digitial (A/D) converter is controlled by the FPGA, and the current sampling frequency is 50 MHz. The sampling frequency can be lowered if required, but it is then important to consider aliasing-effects related to the sampled signal.

2.3 Accuracy, applications and ambitions

There are mainly two different interesting quantities when using the Ultramonit corrosion monitoring system:

1. The absolute thickness based on one measurement.
2. The rate of corrosion based on multiple consecutive measurements.

Currently the “best” practitioners in the field operate with an uncertainty of at least +-0.25 mm for absolute thickness measurements. SensorLink’s ambition is to decrease this to less than +-0.10 mm, a value which in fact is a bit conservative when looking at the results achieved so far.

What is unique with Ultramonit however, is the ability to discover the rate of change, i.e. the *corrosion rate*, with a remarkable precision. That is, even though one operates with an uncertainty of 0.10 mm for the absolute thickness, the relative thickness-deviation between two consecutive measurements can be measured with a much higher precision; the order of magnitude is less than a micrometer (µm). SensorLink’s ambition is to be able to calculate the yearly corrosion rate with an uncertainty of 0.1

mm within a shift of 12 hours. This is from here on denoted *ultra-high precision applications*. A typical yearly corrosion rate of 0.25 mm/year implies a change of 0.34 μm during this shift. It is clear that such tasks require high-quality measurements.

The most typical application, however, will be to fetch data twice a year or so in order to be in control of the corrosion development. If the actual corrosion rate is 0.25 mm/year, one will then expect a change of 0.125 mm from the last measurement. This is considered as the main application of the system.

2.4 Transfer time on existing system

The transfer rate through the inductive coupler is maximum 115200 bit/second. Each ultrasonic trace consists of 2048 samples and the precision is 16 bit/sample. An additional 25% bits are assumed for administrational purposes. The header and checksum add up to 580 bits including bits for link-administration. If there are 150 transducers (which is a realistic number) in one clamp, the total number of bits yields:

$$(2048 * 1.25 * 16 + 580) * 150 bit = 6231000 bit$$

Dividing this number with 115200 bit/second gives 54 seconds. The transfer time for one trace is correspondingly 54/150 s = 0.36 s. Knowing that obtaining the 150 measurements takes about 1 second, we clearly see that the communication link is the system's bottleneck.

3 Source coding theory

Source coding deals with efficient representation of digital data. This thesis is concerned with lossy compression, i.e. the source is not perfectly preserved in the representation. Thus one can say more explicitly that data compression is the representation of a source in digital form with as few bits as possible while maintaining an acceptable loss in fidelity [4]. For audiovisual signals it is common to refer to this acceptable fidelity-loss as *irrelevant information*, a term which is well established in the audiovisual community as non-perceivable information. In this thesis ultrasonic traces are the target of compression. What is irrelevant for this type of data is discussed in section 4.2 and chapter 6.

Performing compression on acoustic material is a difficult task if everything is to be done in one operation. Vector quantization methods based on codebooks, such as the *Generalized Lloyd-Max algorithm* (or equivalently *The Linde-Buzo-Gray algorithm*) [5], are suited for that kind of compression, but are often avoided because of their high computational complexity and delay associated with searching the codebook for the best representation vector. Therefore one typically splits the source coding process in three blocks as illustrated in the schematic figure below.

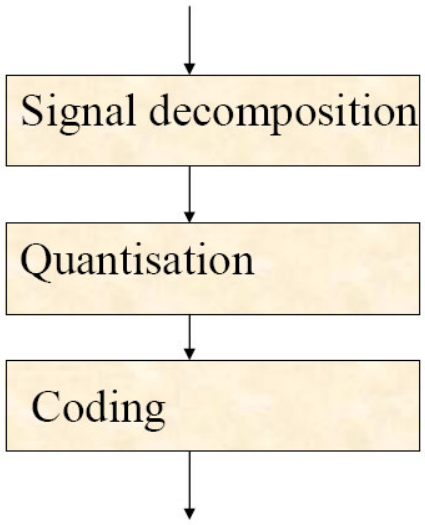


Figure 3 Typical source encoding steps

Signal decomposition is the art of mapping an input signal to another representation where one more easily can exploit the signal characteristics by simple source coding techniques. Widely used decomposition methods are:

- Predictive and differential coding
- Transform coding
- Subband coding

What the above methods have in common is that they aim at *decorrelating* the signal such that the *redundancy* may be taken better advantage of. Predictive and differential coding processes the signal in the time-domain, and makes use of a prediction filter which fits the signal characteristics. The filter coefficients are typically based on

random signal modeling, such as *autoregressive modeling* [6]. The optimum linear predictor in a *mean square error* (mse) sense, is based on the well known Y*oule-Walker* equations [6].

Transform coding is another form of signal decomposition which transforms the input signal vector from the time-domain into another domain, e.g. the frequency domain, where the coefficients are more easily coded. Usually the goal is energy-packing, where most of the signal's energy is located in as few transform coefficients as possible. Furthermore, the transforms are in general *invertible*, i.e. the original signal can entirely be reconstructed from its transform coefficients without loss of information. The *Karhunen-Loeve Transform* [7] is the optimal orthogonal transform based on certain criteria. It is, however, used infrequently as it is dependent on the statistics of the input sequence. Other discrete transforms, such as the *discrete cosine transform* are much more popular due to its signal independence and related fast algorithms.

In s*ubband coding,* which basically is a generalization of transform coding, an analysis filter-bank operates on the input signal to generate a set of narrowband signals each representing a different subband in the frequency domain. Again one aims for energy packing and decorrelation between the subbands. After the filtering process, decimation is usually performed in order to achieve *critical sampling.* Then the number of coefficients is constant before and after filtering, and the order of decimation equals the number of subbands, i.e. if the filter-bank splits the signal into K subbands, each of them is *downsampled* with a factor K. Filtering and downsampling is together called *decimation*. Due to the narrow bandwidth of each subband, information is generally not lost during the downsampling process. The original signal is reconstructed through *upsampling* (where K-1 zeros are inserted between each coefficient) and the use of a synthesis filter-bank, which together is denoted *interpolation*. If the analysis-synthesis scheme has no reconstruction error in the absence of quantization, it is called *perfect reconstruction.* A special case of subband coding based on *Wavelet Packet filter banks* is the signal decomposition method used in this thesis. Wavelets and wavelet packets are further explored in section 3.3.

Referring to Figure 3, the next step after signal decomposition is *quantization*. This step introduces *distortion* and is what makes the compression *lossy*. The task of the quantizer is to approximate each coefficient from the signal decomposition output to its nearest neighbor in a finite set Q. The main reason for using a vector quantizer is that it "sees" statistical dependencies not seen by a scalar quantizer. But since the signal decomposition has decorrelated the original input signal to a certain degree, it is normally sufficient with the much less complex scalar quantizer. Due to its low execution-delay, a *uniform quantizer with dead-zone* is chosen as the quantizer in the proposed source coder. Scalar quantization and basic distortion theory are further explored in section 3.2.

The last compression step is coding, which aims at representing the output of the quantizer with as few bits as possible with information preserving operations. An important aspect of coding is *entropy coding.* The two main classes of entropy coding are universal codes which has no knowledge of the source to be coded, and model-based methods which assumes a known probabilistic model for the source. Examples

from the first category are the *Lempel-Ziv* variants [8] which also are the basis for the popular *zip* programs such as *winzip*. Examples from the latter category are *Huffman coding* and *arithmetic coding* [8]. Due to execution-delay-issues and memory constraints none of the above entropy coders are used in the proposed source coder. A low-complexity coding method called *run-length coding* is implemented though. This is further explored in section 5.6. Another important aspect of coding is to represent a K bit quantizer with no more than K bits. As the computer's memory only stores bytes (i.e. one memory address represents one byte), one has to perform bit manipulation in order to create a data stream where each value is represented with a different number of bits than modulo 8.

After the steps in Figure 3 are performed, the encoded signal has to be decoded in order to regain a distorted version of the original signal. The decoding process includes a binary decoder, an inverse quantizer and a synthesis filter bank which reconstructs the signal.

3.1 Source coding fundamentals for discrete-time signals

This section deals with basic source coding theory.

3.1.1 Source and source alphabet

Consider the *information source,* X, which generates sequences of succeeding *source symbols*, $\{x_n\}$, $n \in \mathbb{N}$. The source symbols can be regarded as realizations of a random process $\{X_n\}$, $n \in \mathbb{N}$. The information source in this thesis is represented by the transducers and the analog to digital converter, which send and receive the ultrasonic signals, and samples and quantizes the incoming data respectively. As the produced data are sampled, the information source is said to be *discrete*. Each of the source symbols is assumed to take one of M values from the finite *alphabet* $A_X = \{a_0, a_1, ..., a_{M-1}\}$, where $a_0 < a_1 < ... < a_{M-1}$.

3.1.2 Information content and entropy

The source X generates each of the symbols in A_X with probabilities $P(X = a_m) = p_m$, m = 0, 1,…, M-1, where $\sum_m p_m = 1$. The *information content* of a_m is expressed as $I_m = -lb(p_m)$ bits where lb(.) is the base 2 logarithmic function due to the binary data-representation. The *entropy* of X is defined as

$$H(X) = E\{I\} = \sum_{m=0}^{M-1} p_m I_m \qquad (3.1).$$

Here E{.} denotes the expectation operator. For a *memoryless* discrete source, Shannon states in [9] that the entropy gives the minimum achievable bitrate (in bits/symbol) for a lossless representation. A source is said to be memoryless if all the succeeding symbols generated by the source are statistically independent.

3.1.3 The entropy rate

Most signals exhibit statistical dependence to some degree though, for which the *entropy rate* gives the bound on the minimum achievable bitrate. The entropy rate is given by [10]

$$B = \lim_{L\to\infty} H(X^{(L)}) \tag{3.2}$$

where $X^{(L)}$ is the L^{th} order *source-extension*. A source-extension refers to the case where the symbols in the new alphabet, $A_{X^{(L)}}$, are represented by all possible combinations of symbol-sequences of length L from the original alphabet, A_X. The entropy rate is always a lower bound on the entropy, i.e. $B \leq H(X)$, with equality for memoryless sources only.

3.1.4 Conditional entropy and mutual information

Assume a channel with input X and output Y. Then the *conditional entropy* H(X|Y) is a measure of the expected uncertainty in X after observing Y. It is defined as

$$H(X\,|\,Y) = -\sum_{x\in A_x}\sum_{y\in A_y} P(X=x, Y=y)\,lb(P(X=x\,|\,Y=y)) \tag{3.3}.$$

Here A_Y is assumed to be the alphabet of the output Y, P(X = x ,Y = y) is the joint probability of X = x and Y = y and P(X = x |Y = y) is the conditional probability of X = x given that Y = y. The *mutual information*, I(X;Y), is a measure of how much information which is shared by X and Y, or put another way, how much information one can gain from X by only observing Y. It is defined as

$$I(X;Y) = H(X) - H(X\,|\,Y) = I(Y;X) \tag{3.4}.$$

3.2 Scalar quantization and basic rate-distortion theory

As discussed in section 3.1.1, the discrete source X generates symbols with values in the interval $[a_0, a_{M-1}]$. A scalar quantization process, Q(.), implies to split this interval into N non-overlapping intervals $[c_k, c_{k+1}]$, k = 0, 1, …, N, where N < M, and map each input symbol with values in the interval $[c_k, c_{k+1}]$ to the representation value $b_k \in [c_k, c_{k+1}]$. The mapping is based on the nearest neighbor principle, i.e. $Q(x) = b_k$ if and only if $d(x, b_k) \leq d(x, b_i)$ for all $k \neq i$. Here $d(.,.)$ represent the *single-symbol distortion.*

The quantizer can be thought of as a new discrete source, Y, which, given the input sequence $\{x_n\}$, generates the sequence of source symbols $\{Q(x_n)\} = \{y_n\}$. Now each y_n takes values from the finite alphabet $A_Y = \{b_0, b_1, ..., b_{N-1}\}$.

Since $N = |A_Y| < |A_X| = M$, the sequence $\{y_n\}$ can obviously be represented by fewer bits than the original sequence $\{x_n\}$. The downside is that quantization is an irreversible operation which looses information. To see this consider the quantized value $Q(x_n) = y_n = b_k$. Given this scalar one can never figure out the original value of x_n. One can only decide which interval it belonged to, namely that $x_n \in [c_k, c_{k+1}]$. The quantizer has thus introduced the single-symbol distortion $d(x_n, y_n)$. In this thesis the single-symbol distortion-measure is represented by the mse, i.e. $d(x_n, y_n) = (x_n - y_n)^2$. Knowing the probability of occurrence of each value in A_X, the expected distortion of a quantized value is given by

$$D_{X,Y} = \sigma_Q^2 = E\{d(X,Y)\} = \sum_{x \in A_X} p_X(x) d(x, Q(x)) = \sum_{x \in A_X} p_X(x)(x - Q(x))^2 \quad (3.5).$$

This is also known as the quantization error variance [11]. Intuitively one aims at minimizing the distortion for a given rate. A classic result in the topic of scalar quantization design, is that the optimal representation level in the interval $[c_k, c_{k+1}]$ with respect to $D_{X,Y}$ conforms with the centroid of the probability density function (pdf) of X in that specific interval [12]. This result is based on the *high-rate approximation which we define as true whenever the input variance is much larger than the distortion*, i.e. when $\sigma_Q^2 << \sigma_X^2$ [13] .

The above results are related to a specific type of coding scheme. If one instead considers the space E of all possible encoders, $\{e_i\}_{i \in \mathbb{Z}}$, assigning the distortion D_{E_i} to encoder e_i, then the minimum achievable rate, R, for a given distortion, D, is related to the mutual information:

$$R(D) = \min_{D_{E_i} \le D, i \in \mathbb{Z}} I(X;Y) \quad (3.6).$$

According to *Shannon's fundamental rate-distortion theorem*, there always exist an encoder which, given the distortion D, can represent the source with a rate arbitrary close to the minimum achievable rate. Typical properties of the rate-distortion function are [14]:

- Convexity. This implies that representing the source with an extra bit will have a larger impact on the distortion the lower the rate is to begin with.
- It is a monotonic decreasing function. Thus one can always represent the source with fewer bits when the distortion increases.

An example of a rate-distortion function is seen in Figure 4.

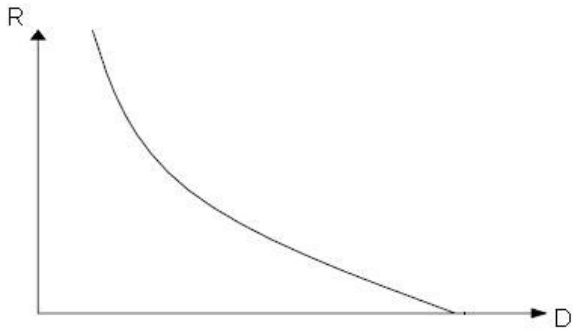


Figure 4 A typical rate-distortion function

3.2.1 Uniform quantization

Uniform quantization is a simple quantization scheme based on equispaced representation levels. Thus, if the coefficient values for a subband represented with R bits/sample are in the interval [-A,A], then the *quantization interval*, Δ, becomes

$$\Delta = \frac{2A}{N} \tag{3.7}.$$

Here N is the number of representation levels given by $N = 2^R$. The distortion is still given by equation (3.5), but if the high-rate approximation holds, the following simple expression gives a good estimate of the quantization noise [15]:

$$\sigma_Q^2 = \frac{\Delta^2}{12} \tag{3.8}.$$

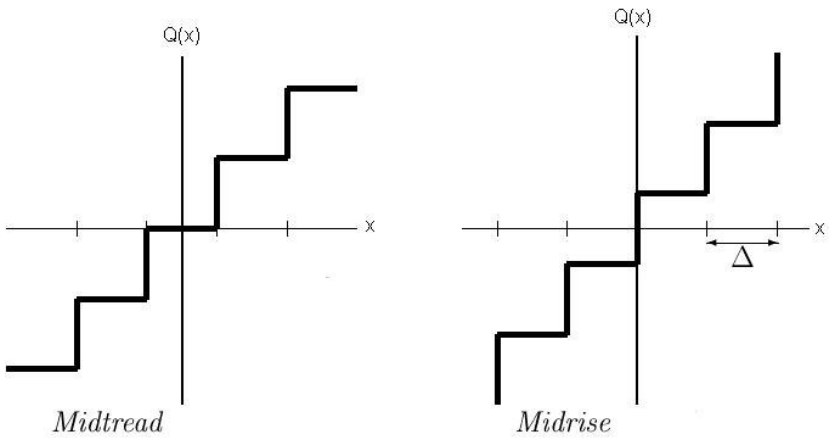


Figure 5 A midtread quantizer to the left and a midrise quantizer to the right. The midtread quantizer has 0 as a representation level, while the midrise quantizer is symmetric around zero.

The uniform quantizer, for which two variants are demonstrated above, is attractive due to its simplicity, but as a matter of fact it is in a way optimal as well: Another approach to minimizing the distortion is to do this indirectly by searching for a

quantizer which minimizes the entropy of the output instead. It can then be proven, again using the high-rate approximation, that the optimal quantizer is uniform [16]. Thus, if one uses uniform quantization together with an efficient entropy coder, this is in fact a better method than the pdf-optimized scheme described earlier.

3.3 Wavelets and subband coding

Subband coding is a signal decomposition method where the signal is subdivided into several frequency bands by the use of a filter bank. In order to maintain the number of coefficients somewhat constant, each subband-vector is critically downsampled after the filtering process. Critical downsampling gives rise to aliasing which must be negligible in the reconstructed signal. *Conjugate mirror filters* (CMFs) are a special class of widely used perfect reconstruction filters which deals with the aliasing problem.

The class of wavelets we will look into in this thesis is entirely characterized by its related filter. It turns out that these filters always fulfill the CMF properties. Thus, the *discrete wavelet transform* (DWT) is actually a subclass of subband coding using CMFs. Wavelets and their related discrete filters have had a growing impact on source coding theory and practice for more than a decade. Today, JPEG2000 [17] is one of the most well-known compression standard where the DWT is applied. Due to the great success and applicability of wavelet theory in source coding, a variant of the DWT denoted *wavelet packet decomposition* is employed as the signal decomposition engine in this thesis. In order to try to understand the favorable properties of wavelet-theory in signal decomposition, a discussion about this theme, and especially about orthogonal wavelets, follows. See also [15] for an excellent introduction to wavelet theory.

3.3.1 Orthogonal wavelets

This section gives a brief introduction to orthogonal wavelets as well as the DWT. For a more thorough discussion, see Appendix B.

3.3.1.1 General theory

An important definition in the theory of wavelet bases is that of *multiresolution approximation* (MRA) [18]. Note that some texts use the term multiresolution analysis instead. Herein the term approximation fits better, due to its relation to compression. Recall that lossy compression definitely is a kind of approximation.

With MRA one can approximate a function f at a resolution 2^{-j}. This approximation is a kind of *local average* of f over intervals proportional to the scale 2^{j}. As we want orthogonal wavelet bases, the approximation of f at the resolution 2^{-j} is an *orthogonal projection* onto the *approximation space* $\mathbf{V}_j$. The projection is denoted $P_{\mathbf{V}_j} f = f_j$.

A favorable property of the MRA is that $\lim_{j \to -\infty} \mathbf{V}_j = \mathbf{L}^2(\mathbb{R})$ where $\mathbf{L}^2(\mathbb{R})$ is the space of square integrable functions. Thus any function $f \in \mathbf{L}^2(\mathbb{R})$ can be approximated to an arbitrary precision. Furthermore, another property of the MRA defines the relation

between the spaces $\{\mathbf{V}_j\}_{j\in\mathbb{Z}}$, namely that $\forall j \in \mathbb{Z}, \mathbf{V}_{j+1} \subset \mathbf{V}_j$. That means that an approximation in $\mathbf{V}_j$ contains all information for computing an approximation at a coarser resolution. The implication of this is that if one wants to approximate f with respect to $\mathbf{V}_{j+1}$, and already has the approximation f_j, then $P_{\mathbf{V}_{j+1}} f = P_{\mathbf{V}_{j+1}} f_j$. This is important for the DWT where one successively decomposes each approximation $P_{\mathbf{V}_j} f$ into coarser approximations $P_{\mathbf{V}_{j+1}} f$.

The space $\mathbf{V}_j$ is spanned by the *orthonormal basis functions* $\{\frac{1}{\sqrt{2^j}}\varphi(\frac{t-2^j n}{2^j})\}_{n\in\mathbb{Z}}$ where the function φ is called the scaling function. Since $\mathbf{V}_1 \subset \mathbf{V}_0$ (note that j is arbitrary; 0 and 1 is just chosen for convenience) the scaling function of $\mathbf{V}_1$ can be decomposed into a weighted sum of the orthonormal basis functions of $\mathbf{V}_0$:

$$\frac{1}{\sqrt{2}}\varphi(\frac{t}{2}) = \sum_{n=-\infty}^{\infty} h_n \varphi(t-n) \tag{3.9}.$$

Equation (3.9) is called the *scaling function in time-domain.* The sequence h_n is interpreted as a filter; it is actually the DWT's low-pass filter! Note that the filter is independent of the scale. It can be demonstrated that this filter has the desired CMF property (see [19], [20]) which grants perfect reconstruction for finite impulse response (FIR) filters:

$$|H(\omega)|^2 + |H(\omega+\pi)|^2 = 2 \tag{3.10}.$$

Here $H(\omega)$ is the discrete-time Fourier transform of h_n, defined in equation (10.3).

Since $\mathbf{V}_j \subset \mathbf{V}_{j+1}$, there must exist a space $\mathbf{W}_{j+1}$ orthogonal to $\mathbf{V}_{j+1}$ such that $\mathbf{V}_{j+1} \oplus \mathbf{W}_{j+1} = \mathbf{V}_j$. Here $\oplus$ is the orthogonal direct sum operator [21]. The space $\mathbf{W}_{j+1}$ can be interpreted as the space which contains the difference between the approximations on scale 2^j and 2^{j+1}. Let us denote the spaces $\{\mathbf{W}_j\}_{j\in\mathbb{Z}}$ for detail spaces. Similarly as for the scaling function and approximation spaces, the space $\mathbf{W}_j$ is spanned by the orthonormal basis functions $\{\frac{1}{\sqrt{2^j}}\psi(\frac{t-2^j n}{2^j})\}_{n\in\mathbb{Z}}$ where ψ is denoted the wavelet function. Since $\mathbf{W}_0$ clearly is a subspace of $\mathbf{V}_{-1}$ one can express $\psi(t)$ as a series of orthonormal basis functions:

$$\psi(t) = \sum_{n=-\infty}^{\infty} g_n \sqrt{2}\varphi(2t-n) \tag{3.11}.$$

The sequence $\{g_n\}$ is just like $\{h_n\}$ interpreted as a filter, and this time as the DWT's high-pass filter. Note the similarity between equation (3.9) and (3.11). The CMF property also holds for this filter:

$$|G(\omega)|^2 + |G(\omega+\pi)|^2 = 2 \tag{3.12}.$$

An important relation which relates orthogonality between $\mathbf{W}_j$ and $\mathbf{V}_j$ to the filters *H(ω)* and *G(ω)* is given by:

$$G(\omega)H^*(\omega)+G(\omega+\pi)H^*(\omega+\pi)=0 \qquad (3.13).$$

The * - superscript denotes the complex conjugate operator. By choosing

$$G(\omega)=e^{-j\omega}H^*(\omega+\pi) \qquad (3.14)$$

both (3.12) and (3.13) are fulfilled. Thus we see that the filter *G(ω)* is entirely characterized by *H(ω)*. Taking the inverse Fourier transform of equation (3.14) yields:

$$g_n=(-1)^{1-n}h_{1-n} \qquad (3.15).$$

and we have the relation (3.14) in time-domain. To get a more practical understanding on the relations between MRA and the filters *H(ω)* and *G(ω),* see the example in Appendix B.

3.3.1.2 The discrete wavelet transform

Now we are ready to present the DWT. Recall that MRA is about approximating a function on different resolutions 2^{-j}. By employing the wavelet-function one can also keep the detailed information which is lost during the approximation from $\mathbf{V}_j$ to $\mathbf{V}_{j+1}$. The approximations- and detail coefficients on the resolution 2^{-j-1} are obtained by orthogonal projections down to the spaces $\mathbf{V}_{j+1}$ and $\mathbf{W}_{j+1}$ respectively. By denoting the approximation- and detail coefficients at resolution 2^{-j+1} for $a_{j+1}(n)$ and $d_{j+1}(n)$ respectively, we have:

$$\begin{aligned} a_{j+1}(p) &=< f,\varphi_{j+1,p} > \\ d_{j+1}(p) &=< f,\psi_{j+1,p} > \end{aligned} \qquad (3.16).$$

By expanding the right-hand sides of equation (3.16) one obtains after some calculations (see Appendix B):

$$\begin{aligned} a_{j+1}(p) &= \sum_{n=-\infty}^{\infty} h_{n-2p}a_j(n) = (a_j * \bar{h})(2p) \\ d_{j+1}(p) &= \sum_{n=-\infty}^{\infty} g_{n-2p}a_j(n) = (a_j * \bar{g})(2p) \end{aligned} \qquad (3.17).$$

Here * is the convolution operator (and not the complex conjugate as earlier), $\bar{h}_n = h_{-n}$ and $\bar{g}_n = g_{-n}$. Equation (3.17) defines the DWT (also called the fast wavelet transform) and it obviously is a convolution. What is remarkable about this result is that the projection onto the spaces $\mathbf{V}_j$ and $\mathbf{W}_j$ turns out to be a filtering process which can be implemented very effectively on computers in either software or hardware. Looking more closely at (3.17) we see that the equation defines downsampling with a factor 2 as well as filtering, so it actually defines a decimation process.

Using the fact that $\mathbf{V}_{j+1} \oplus \mathbf{W}_{j+1} = \mathbf{V}_j$, we obtain, after some calculation, the following expression at the reconstruction:

$$a_j(p) = \sum_{n=-\infty}^{\infty} h_{p-2n} a_{j+1}(n) + \sum_{n=-\infty}^{\infty} g_{p-2n} d_{j+1}(n) \\ = (\hat{a}_{j+1} * h)(p) + (\hat{d}_{j+1} * g)(p) \quad (3.18).$$

$\hat{a}_{j+1}$ and $\hat{d}_{j+1}$ is the upsampled versions of a_{j+1} and d_{j+1} respectively. Equation (3.18) defines the inverse DWT (IDWT). The equation also shows that the IDWT is an interpolation process, i.e. it upsamples with 2 before filtering. Figure 6 shows a schematic overview of both the DWT and the IDWT.

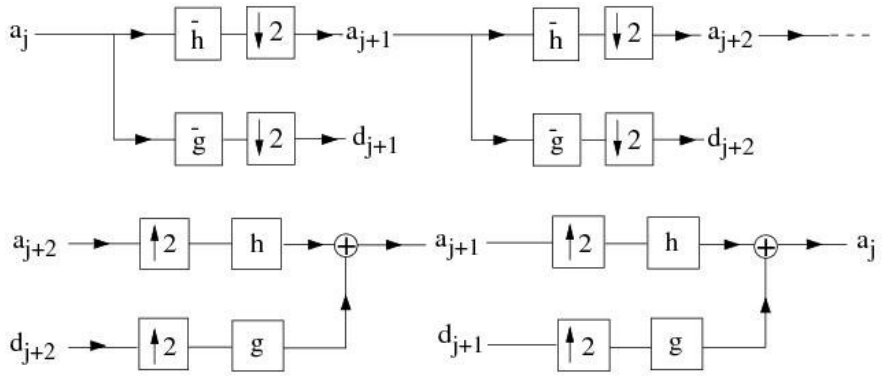


Figure 6 The DWT above and the IDWT below. The boxes with arrows pointing up and down denote upsampling and downsampling with a factor 2 respectively. We see that the filters used in the reconstruction are the time-reversed versions of the analysis-filters.

Multiple levels or scales of the DWT are, as seen in the above figure, made by repeating the filtering and decimation process on the lowpass branch outputs only. The process is typically carried out for a finite number of levels K, and the resulting sets of coefficients (K sets of detail coefficients and 1 set of approximation coefficients) is denoted the *wavelet representation* of the signal.

Let's summarize the most essential so far:

- With MRA, one can approximate any input function in $\mathbf{L}^2(\mathbb{R})$ to an arbitrary precision.
- For each successive approximation there exists a detail space which contains the difference of those approximations. Thus one never looses any information.
- The projections down to the approximations- and detail spaces can be interpreted in terms of digital filters.
- These filters fulfill the CMF property which grants perfect reconstruction.
- The DWT/IDWT can be efficiently implemented in either SW or HW. The DWT output is multiple sets of detail-coefficients (one set for each scale) and one set of approximation-coefficient. This is denoted the wavelet representation of the input signal.

3.3.1.3 Orthogonal wavelet properties

Three important wavelet properties are:

- Size of support.
- Vanishing moments.
- Regularity.

The size of support of the wavelet is directly related to the support of the filters $\{g_n\}$ and $\{h_n\}$. The smaller the support of the filters, the faster the DWT is performed on a processing unit. It is therefore desirable that the filters have few non-zero coefficients.

A wavelet ψ has p vanishing moments if and only if it is orthogonal to all polynomials of degree less than p, i.e. $\int_{\mathbb{R}} t^k \psi(t)dt = 0, 0 \le k < p$. It is then denoted a wavelet of order p. If an input signal can be approximated by a k-order Taylor polynomial in a small interval, the detail-coefficients produced by the DWT will then have small amplitudes in that interval. As our goal is to produce as few large coefficients as possible (energy packing), this is a desirable property.

A theorem due to Daubechies [22] states that *H(ω)* must have p zeros at $\omega = \pi$ (i.e. *H(ω)* and its first p-1 derivatives are zero at $\omega = \pi$) in order to ensure that ψ has p vanishing moments. Such filters have at least 2p non-zero coefficients. A class of wavelet filters denoted Daubechies filters has 2p non-zero coefficients. Thus they are optimal in the sense that they have minimum size support for a given order of vanishing moments. The Daubechies scaling functions and wavelets are presented for $p \in \{1, 2, 3\}$ in Figure 42.

Regularity is related to the time vs. frequency localization of the wavelet. Wavelets with a high degree of regularity have in general better frequency localization. For Daubechies wavelets, the regularity increases with the number of vanishing moments. It is worth noting that the highly irregular Haar wavelet is a Daubechies wavelet of order 1. In many applications, e.g. images, regularity is important because it makes the quantization error less visible (the error is "smoothened").

As the vanishing moments are related to the approximation abilities, they are important in the analysis. Regularity is more important in the synthesis-phase due to smoothing of the quantization errors.

3.3.2 The boundary problem and extension to biorthogonal wavelets

In the discussion about orthogonal wavelets and the DWT, it has been assumed that the input signal has infinite support. For all practical purposes, the input signal is of finite length. When filtering the boundaries of a finite signal, the input signal must be extended in one way or another, or boundary filters [23] must be applied.

There are two common problems when dealing with boundaries:

- Boundary artifacts.
- Coefficient expansion.

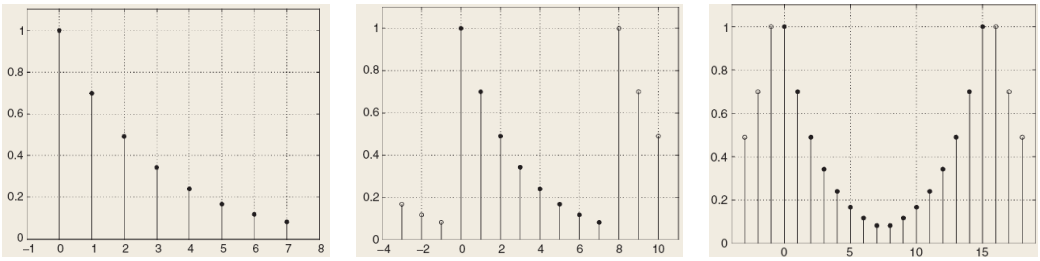
The boundary artifacts arise from artificial discontinuities created after certain signal extension-methods. Thus one obtains large coefficients which demands many bits in the bit-allocation process for reconstruction purposes, but which contains little or no relevant information of the source itself.

Coefficient expansion comes from the filtering process, and the problem will increase with an increasing number of decomposition levels. The contradiction of expanding the number of coefficients when one aims at compression, is of course disadvantageous. An expansion will moreover lead to additional calculations, increasing the processing-time.

A boundary filter, which is a somewhat complex matter, would have dealt with both the problems mentioned above, but in this thesis signal extension is applied. Typical signal extensions are:

- Zero-padding. This method assumes that the signal is zero outside the original support. This is the simplest form of signal extension, and does not deal with any of the abovementioned problems.
- Periodization. This is in practice the same as circular convolution and it deals with the expansion problem. Boundary artifacts are still created though.
- Symmetrization. This deals with the artifacts problem because it guarantees continuity as long as the original signal is continuous. Symmetrization still expands the number of coefficients for orthogonal wavelets, but it indeed keeps it constant for a class of wavelets denoted *biorthogonal wavelets* as long as they are symmetric or anti-symmetric.

The latter to types of signal extension are demonstrated in Figure 7.



a) b) c)

Figure 7 Examples of signal extensions. Original signal in a), periodic extension in b) and symmetric extension in c).

To deal with both signal extension and boarder artifacts, one thus has to employ symmetrization together with symmetric or asymmetric biorthogonal wavelet filters. While only two different filters are applied in the orthogonal case, this must be extended to 4 when using biorthogonal filters. A brief discussion about biorthogonal wavelets are found in Appendix C.

An equivalent efficient algorithm as described in section 3.3.1.2 also exists for biorthogonal wavelet-filters, but now with 4 different filters instead of 2. This is shown in the figure below.

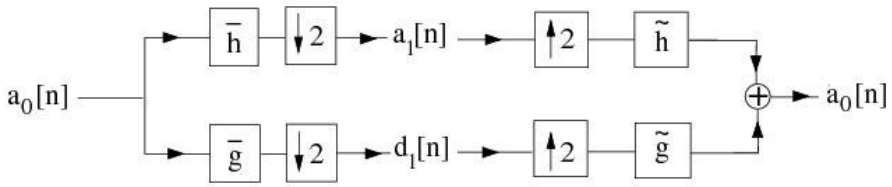


Figure 8 One level decomposition and reconstruction with the biorthogonal filters $\mathbf{h}, \mathbf{g}, \tilde{\mathbf{h}}$ and $\tilde{\mathbf{g}}$.

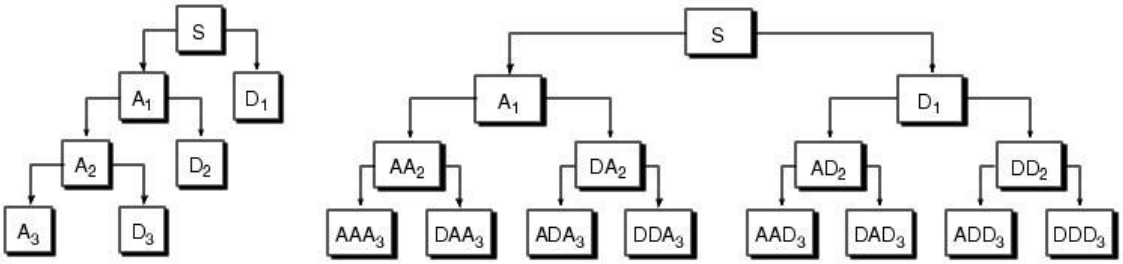
Biorthogonality gives more freedom in the design of wavelets/filters, and in general it can be shown, at least for image coding, that an optimal biorthogonal filter performs better than an optimal orthogonal filter [24]. Some of the beneficial properties are listed below:

- Opportunities of creating symmetric/antisymmetric linear-phase filters.
- Avoid coefficient-expansion without using boarder filters.
- One can concentrate useful properties for analysis (such as vanishing moments) in the analysis filter, and useful properties for the synthesis (such as regularity) in the synthesis filter.

An example of biorthogonal wavelets and scaling functions is shown Appendix C.

3.3.3 Extension to wavelet packets

Instead of dividing only the approximation spaces $\mathbf{V}_j$, it can be proven that one also can divide the detail space $\mathbf{W}_j$ in two new spaces and derive new bases [25]. By using the same wavelet filters as discussed in the last to sections, one then obtains uniform subband-coding instead of the dyadic subband-coding obtained with the traditional DWT. This is denoted *wavelet-packet* (WP) analysis. With this approach one obtains a binary tree of spaces, where each space (or node) in the tree is denoted a WP. A comparison between the 3-level wavelet decomposition and the 3-level WP decomposition of the input signal s is demonstrated in Figure 9. The letter A refers to approximation space and low-pass filtering, while D refers to detail space and high-pass filtering. The succeeding letters in b) refers to the order of which the filters are applied, e.g. DDA_3 means that the signal first has been decimated with the low-pass filter and then twice decimated with the high-pass filter. For convenience one often denotes the nodes in terms of their parent-child relationship. AD_2 is e.g. the parent to its left child AAD_3 as well as to its right child DAD_3. Moreover, the node S is called the root node.



a) b)

Figure 9 A 3-level wavelet decomposition in a) vs. a 3-level WP decomposition in b).

The WP filter-bank decomposition and reconstruction, with the orthogonal wavelet filters $H(\omega)$ and $G(\omega)$, are shown in Figure 10.

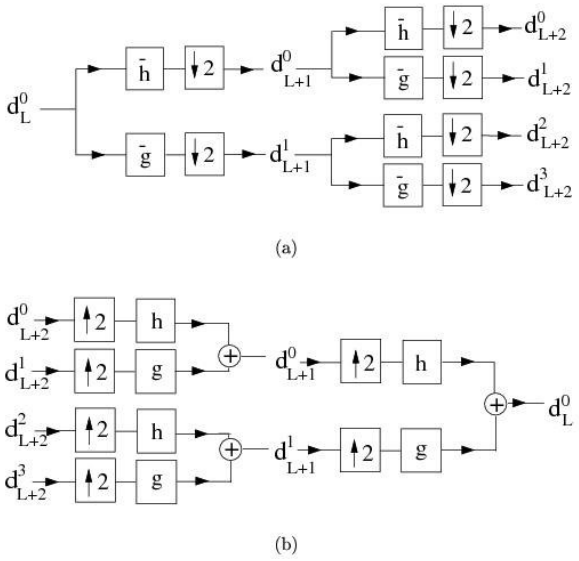


Figure 10 WP filter-bank decomposition a) and reconstruction b).

3.3.3.1 Admissible tree

An admissible WP-tree is any binary tree where each node has either 0 or 2 children. In such a tree the original signal can be represented by the leaf-nodes. If we refer to Figure 9 we can see that the wavelet representation in a) is only one of many representations of the root node S. The wavelet representation $S = D_1 + D_2 + D_3 + A_3$ in a) is equivalent to selecting the following nodes from the WP-tree $S = D_1 + DA_2 + DAA_3 + AAA_3$. While there are three different representations of S in Figure 9 a), namely $S = D_1 + A_1 = D_1 + D_2 + A_2 = D_1 + D_2 + D_3 + A_3$, there are 25 representations in b). Uniform subband-coding is obtained in b) if one represents the signal with the 8 nodes on level 3. Each admissible tree-representation yields of course perfect reconstruction.

In section 5.4.1 we will see that admissible trees play an important role in one of the source coding algorithms.

4 Signal characteristics

This section first gives an explanation of the traces in order to understand which part of the signal which is most significant for our purposes. In the subsequent section, signal properties and irrelevant information regarding the ultrasonic traces are discussed.

4.1 Explanation of the ultrasonic traces

The corrosion experiments which produced the ultrasonic traces used in this thesis had the simple setup shown in Figure 11. Corrosion on the steel's inner wall was induced by means of impressed current using electric power supplies. By varying the electric power, one can change the corrosion rate. In this thesis one low-rate and one high-rate corrosion experiment are examined.

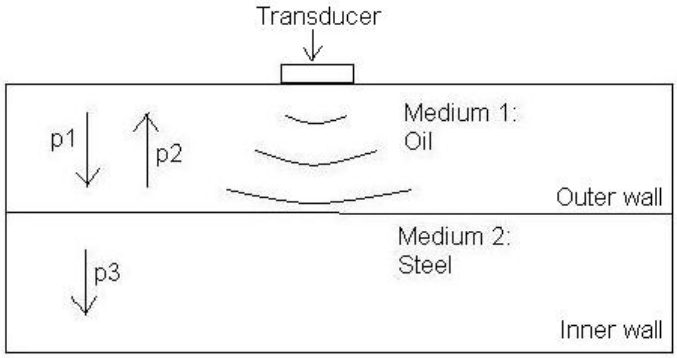


Figure 11 Corrosion experiment.

When the acoustic pulse is transmitted from the transducer, it has to propagate through a layer of oil-based fluid before it reaches the outer wall. At such interfaces between different media, reflection and transmission occur, i.e. some of the acoustic energy is reflected at the interface, while some of it passes through the interface and propagates into the new medium. This is illustrated by the labels p1-p3 in the above figure where p1 is the incoming acoustic pulse, p2 is the reflected pulse while p3 is the transmitted pulse through the interface. Now fractions of the acoustic energy propagate back and forth in both media, and each time they reach the interface, reflection and transmission occur. (Note that we are only concerned with the longitudinal primary waves, also known as P-waves, in this thesis.) A typical ultrasonic trace, here labeled with numbers, is presented in Figure 12. The deflection at label 1 represents the first reflection from the outer wall, i.e. the acoustic wave has propagated from the transducer to the interface and back to the transducer again. The labels 2, 3 and 4 represent reflections from the inner wall where the acoustic wave has propagated once, twice and three times back and forth in the steel respectively before moving back to the transducer.

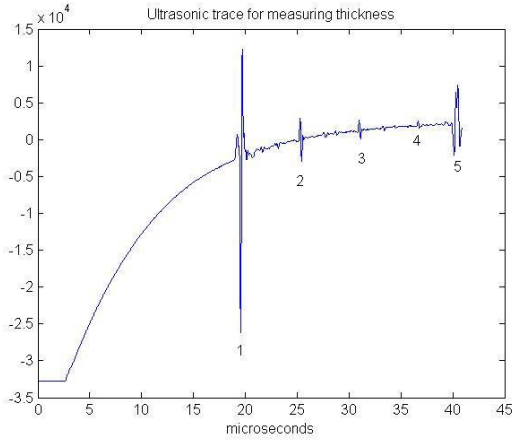


Figure 12 A typical ultrasonic trace.

Label number five represents the pulse which has traveled twice back and forth in oil. The section from label 1 to label 5 is generally the interval of interest. Since the time interval between label 1 and label 2 is the time it takes for the acoustic pulse to move once back and forth in the steel, this information is used to calculate the thickness, which is given by the following self-explaining formula:

$$thickness = \frac{time_{label1 \rightarrow label2} acoustic\_velocity_{steel}}{2} \qquad (4.1).$$

Note that the layers of oil and steel can be interpreted as a *transmission channel* where the *channel characteristics* will vary due to parameters such as temperature, oil properties, steel properties, corrosion etc.

4.2 Signal properties and irrelevant information

The transmitted pulse and the channel decide the form of the received signal. In the search for irrelevant information, we examine both time- and Fourier domain in order to reduce the number of samples. This will both increase the compression ratio and decrease the total processing time. This section presents the transducer signature as well as time-/frequency domain plots of different received signals.

4.2.1 Transducer signature

Transducers are characterized by their signature which demonstrates the time- and frequency content of the transmitted pulse. It is clear that the signature will have a significant impact on a received signal, and that their frequency content to a large extent will coincide. In the abovementioned experiments, one of the main alternatives for the final choice of transducers in the Ultramonit project, a 5 MHz Telsonic 6mm transducer [26], is employed. Its signature is presented in the figure below.

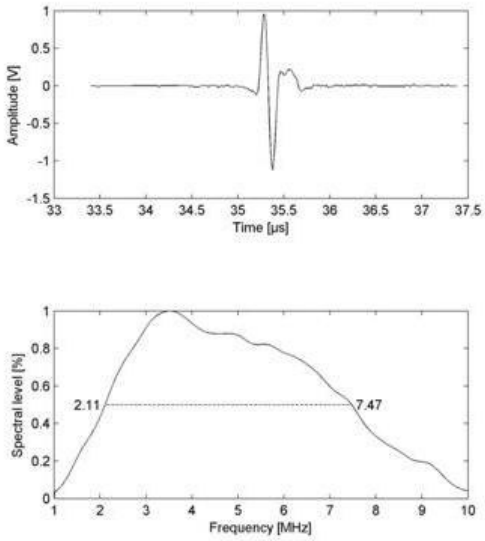


Figure 13 The signature of the Telsonic 5 MHz transducer.

Note that the transmitter pulses have a bandpass characteristic, which implies that any large LP-components in a received signal may be irrelevant information.

4.2.2 Received signal characteristics

The sampling frequency for the A/D-converter is 50 MHz. According to Nyquist's sampling theorem (see e.g. [27]), aliasing is avoided if the original signal only contains frequencies below 25 MHz. Transforming the trace in Figure 12 into the Fourier domain gives rise to the following figure:

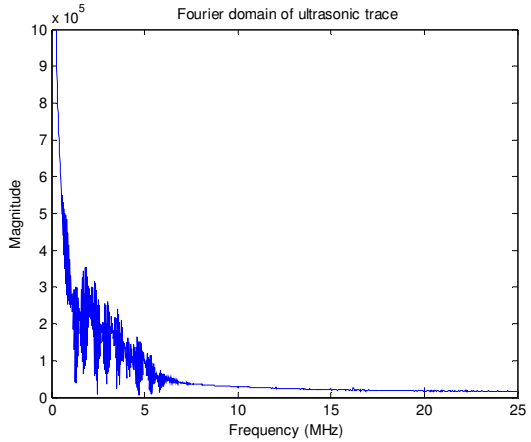
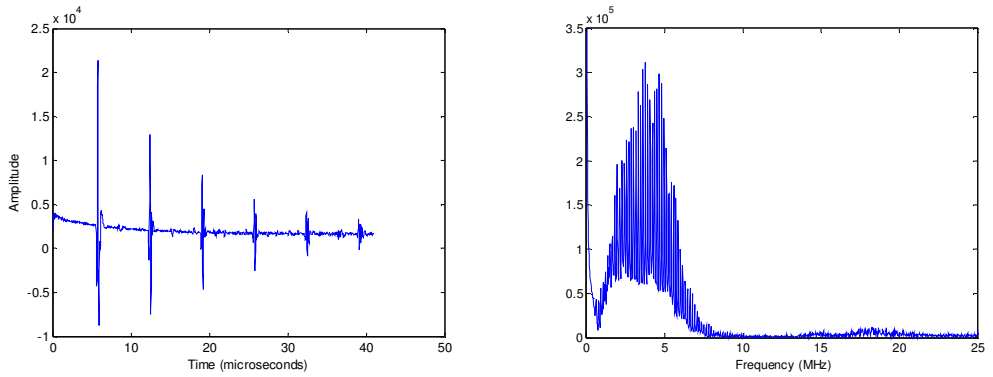


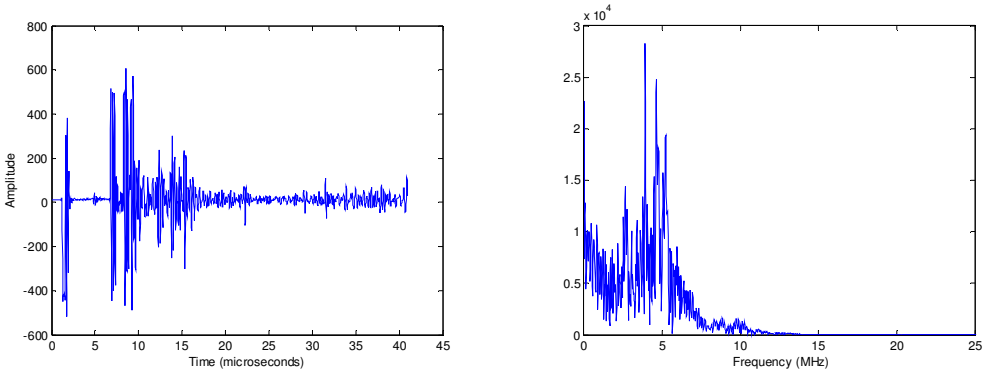
Figure 14 The Fourier domain of the signal shows that there is very little information in the upper frequencies. Note that the y-axis is limited in order to see the details of the frequencies above DC.

Other time-/frequency characteristics are given below. :



a) b)

Figure 15 Time- and frequency domain characteristics in a) and b) respectively.



a) b)

Figure 16 Time- and frequency domain characteristics in a) and b) respectively. Note that in this experiment, a 3.5 MHz Panametrics transducer is used.

Note that the traces employed in the compression tests are similar to the trace in Figure 12. The traces in Figure 15 and Figure 16 are taken from experiments in an earlier phase of the Ultramonit development project, but are included herein to show that one can make some generalizations even though the channel characteristics, transducer type etc. changes. Figure 16 is from an experiment where a type of corrosion denoted pitting is simulated and we see that a lot of noise is added to the received signal. If we look at the figures above (from Figure 12 to Figure 16) the following conclusions can be drawn:

- Most of the information lies below 10 MHz for all channels as well as for the transducer signature, even for the noisy signal in Figure 16 a). It may seem that the signal is highly oversampled, but a result presented in Appendix I shows that the sampling frequency should not be reduced without an analog anti-aliasing filter.
- By comparing the Fourier-domain transducer-signature in Figure 13 with Figure 14, it seems that the DC-component in Figure 12 is irrelevant for the pulses which are supposed to give us information about the thickness.
- The third thing to note is that the first 18 μs of the recorded signal in Figure 12 is totally irrelevant for our purposes.

These conclusions will have an impact on the encoding scheme, and is further discussed in section 5.2.

5 The encoding scheme

The encoding scheme is shown in the block diagram in Figure 17. The encoder first preprocesses the input signal before decomposing the signal into a complete four level WP-tree using the algorithm described in section 3.3.3. The choice of wavelet filters employed in the analysis is briefly discussed in section 5.3. Further, as the LP-band is claimed to be irrelevant, the coefficients for this node are not even calculated. This saves some processing-time, and for compression purposes this also removes 1/32 of the original number of samples. Both the preprocessing and the zero allocation of the LP-band is explained in section 5.2.

After the signal decomposition, bits are allocated to the different nodes according to either of the two algorithms described in section 5.4. One of the algorithms is quite simple while the other is slightly more complex. The reason for this choice is to compare algorithms with different complexity both with respect to the processing time and compression ratio. Recall that we have limited processing resources.

The quantizers are chosen to be uniform with a dead-zone as explained in section 5.5. Such quantizers are easy to implement and they require few computations. Finally run-length encoding and binary coding is performed. Our hope is that the threshold-effect of the dead-zone improves the run-length encoding performance by creating long zero-runs. The run-length encoding scheme is discussed in section 5.6.

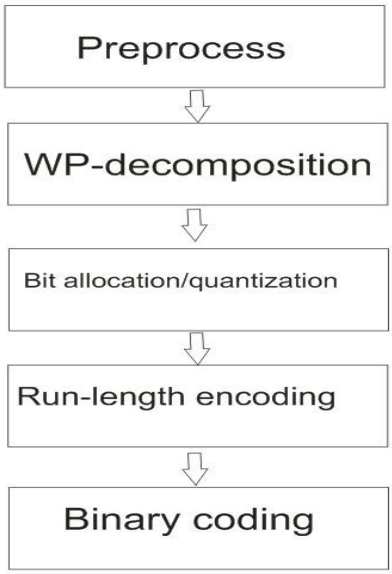


Figure 17 Block diagram of encoder

5.1 Implementation issues

5.1.1 Implementation stages

The implementation process has been performed in three stages:

1. Implementation of critical functions in MATLAB. The critical functions are identified as the analysis and synthesis of the WP-tree and the run-length encoder/decoder.

2. Implement the full encoding/decoding scheme on a standard PC with the C programming language.
3. Implement the encoding scheme onto the MicroBlaze microcontroller, still using C.

On each stage the correctness of the implementation has been verified against the previous stage. The MATLAB Wavelet Toolbox has also been employed for verification, especially in the first stage.

5.1.2 Memory constraints

Implementation on a microcontroller involves heavy memory constraints compared to programming on an ordinary PC. In our case a "memory-pool" of 32 KB block-RAM is directly accessed by the C-compiler (it is actually a total of 64 KB, but 32 of them are reserved VHDL-code.) When compiling the C-code, the compiler calculates the need of memory and returns an error if the limit is exceeded. C-code for administrating the data flow needs approximately 8 KB which leaves about 24 KB for our application. Most functions from the standard C-library, such as input/output- and math-functions, demand so much memory that they should not be employed at all. Another limitation is that of dynamic memory-allocation; this is in practice no option, again due to memory constraints as well as inefficient use of the already limited amount of resources.

We do need more than 32 KB, so how do we solve this? Recall from section 2.2 that there is available 1 MB SRAM on each SAM-module, but the compiler will not use this memory as its "pool" such as is the case for an ordinary PC. The solution is to force the program to put variables on specified addresses in the SRAM. This is denoted static memory allocation due to the programmer's exact knowledge and control of the memory locations used. Typical variables to put in the SRAM are the nodes of the WP-tree and their associated coefficients and quantized values.

5.1.3 Real-time constraints

The real-time nature of the source coding introduces constraints with respect to the processing time as well; the processing time for one trace should not exceed the transfer time for the corresponding trace. But simultaneously, one wants as large compression ratio for a given fidelity as possible, which in general implies higher complexity. As higher complexity in turn leads to higher processing time, we see that there is an intricate relation between transfer time, processing time and compression ratio for a given fidelity. This is further discussed in section 6.1 and 6.2.

To achieve a low processing time, low-complexity components, such as the uniform quantizer with dead-zone and the run-length scheme, are selected in the encoder. Further, a simple bit allocation scheme is chosen. There is a contradiction between the use of uniform quantization and bit-allocation (recall that uniform quantization is optimal with entropy coding while bit allocation is optimal with pdf-optimized quantizers,) but the combination of using a dead-zone together with the run-length encoder yields good results while maintaining a low complexity. This is due to the nonlinear approximation properties of the DWT/WP-decomposition [28]. Note that a

more advanced bit-allocation scheme is implemented as well for comparison purposes, both with respect to fidelity, compression ratio and processing time.

The last step in achieving a low processing time is to use integer arithmetic only. This reduces the precision of the filter coefficients, which theoretically is irrational numbers, to 4 decimals due to overflow-issues. The precision is also reduced in the quantization-process. Even though there is a dedicated unit for floating point operations on the MicroBlaze, tests have shown that using such an approach is too time-consuming for our application.

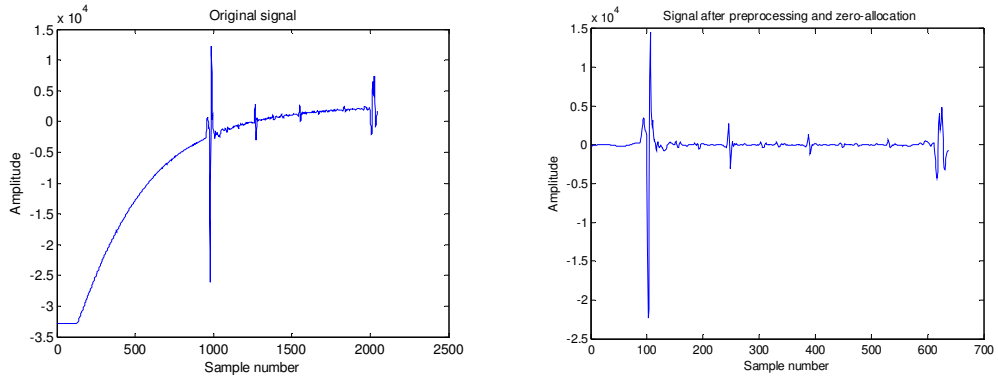
5.2 Preprocessing and zero-allocation

The signal characteristics identified in section 4.2 and the related conclusions can be exploited for reducing the number of samples before signal decomposition. This is important due to the limited HW-resources and constraints on the processing time. The following two preprocessing steps are therefore implemented in the encoder:

1. Identify the first reflection from the outer wall and cut the signal 300 samples in front of that.
2. Decimate the signal with a factor of 2, by first using the simple Haar filter. The Haar filter then functions as an anti-aliasing filter.

By performing the abovementioned operations on the signal in Figure 12, the initial number of samples is reduced from 2048 to 656 which is a factor of 3.

After the preprocessing, the signal is decomposed into subbands. In order to remove the DC-component and gain better compression, all the coefficients in the lowest frequency band on the maximum level is set to zero. This is denoted *zero-allocation*. In contrast to video and images, where the lowest frequency band is the most important, we claim here that it is irrelevant information. In Figure 18 the original signal and the signal after preprocessing and zero-allocation is presented for comparison purposes.



a) b)

Figure 18 Original signal in a), and signal after prepocessing and zero allocation in b). Note that the number of samples is reduced from 2048 to 656.

As a matter of fact, the DC-component is filtered away before processing the thickness anyway. This is due to the fact that such processing is much easier if the

mean of the samples is zero. How the preprocessing and zero-allocation affect the original traces is discussed in the results section.

5.3 Filters

Two different filters are tested in this thesis:

- An orthonormal Daubechies filter of order 5, resulting in 10 taps for both the analysis and synthesis filter. This filter is from here on denoted the 10/10-filter.
- A biorthogonal 7 tap/9 tap filter. That is, in the analysis phase, the LP-filter has 9 taps while the HP-filter has 7 taps, while it is opposite in the synthesis phase. The filter is equivalent to that used in the JPEG2000 scheme. This filter is from here on denoted the 7/9-filter. The scaling function and wavelets resulting from this filter are presented in Appendix C, Figure 43.

The frequency response as well as filter coefficients for both filters are presented in Appendix D.

5.3.1 Why use wavelet filters?

There exist many perfect reconstruction filter banks which exhibits the CMF property without characterizing a wavelet. Some audio CMFs are e.g. optimized for a minimal transition band instead of dealing with the wavelet properties such as $H(0) = \sqrt{2}$ and the zeros at $\omega = \pi$. So why use wavelet filters? In addition to the already well documented approximation properties, equation (10.12) in Appendix B demonstrates that an infinite product of such wavelet filters indeed yield a basis of $\mathbf{L}^2(\mathbb{R})$. This is not the case for other CMFs. The result is that many cascades of the latter class of filters may exhibit uncontrolled behavior. This will not happen for wavelet filters and they are therefore more robust.

This thesis does not claim whether or not the selected filter is the optimal filter for decomposition of the ultrasonic traces, and it is not in the scope of this text to find such an optimal filter either. The author believes though, that the chosen wavelet filter-banks have properties of robustness, signal-approximation and energy-packing which are satisfactory for the signal-decomposition of our signal.

5.4 Bit allocation algorithms

Two bit allocation algorithms are implemented and tested. The related theory is presented in the following two sections.

5.4.1 Best wavelet packet bases in a rate-distortion sense

The idea of the algorithm is to decompose the signal in a complete WP-tree and then to find the “best” admissible tree in a rate-distortion sense. By assigning a set of quantizes and the corresponding distortions to each node in the tree, and thereafter selecting the quantizers and the nodes which gives the least distortion, this algorithm is in a sense optimal. Recall that admissible trees are explained in section 3.3.3.1. The algorithm is entirely based on the ideas in [29].

Note that the algorithm, which we from here on refer to as the BWP-algorithm, is based on the fact that the mse distortion won't change after reconstructing as long as orthogonal projection is used (see Appendix E). This implies that the quantization error made in the wavelet domain equals the quantization error for the reconstructed signal if orthogonal wavelet filters are applied. It is also important not to mix the optimal rate-distortion function over the set of all possible encoders, discussed in section 3.2, with the rate-distortion curves discussed here. In this section, the rate-distortion is always related to the given set of quantizers/entropy coders, and not in any way optimal with respect to all possible encoders.

After the WP-decomposition of depth P, the WP-tree consists of $\sum_{i=0}^{P} 2^i$ nodes. Let us use the following notation:

- T: the complete WP-tree.
- $S \subseteq T$: Set of admissible trees.
- $\tilde{S}$: Set of leaf nodes of S. Leaf nodes are also called terminal nodes.
- n_i : Node number *i*
- $\mathbf{q}(n_i)$: Set of quantizers for node *n*. Herein, the same set of quantizers are chosen for each node.
- $\mathbf{Q}(\tilde{S})$: Set of all possible collections of quantizers for $\tilde{S}$, i.e. $\mathbf{Q}(\tilde{S}) = \{\mathbf{q}(n_1), \mathbf{q}(n_2), ..., \mathbf{q}(n_L)\}$ where $\tilde{S} = \{n_1, n_2, ..., n_L\}$. Note that the choice of quantizers are completely arbitrary, and it is also possible to include entropy coding.
- $D_q(n_i), R_q(n_i)$: Distortion and the corresponding rate for node n_i when using the quantizer $q \in \mathbf{q}(n_i)$.
- $D_Q(S), R_Q(S)$: Total distortion and rate for the leaf nodes in S when using the quantizers $Q \in \mathbf{Q}(\tilde{S})$. Clearly $D_Q(S) = \sum_{n_i \in S} D_q(n_i), R_Q(S) = \sum_{n_i \in S} R_q(n_i)$.
- R_{Budget} : Total number of bits available.

The problem of minimizing the distortion can be formulated as a constrained optimization problem:

$$\min_{S,Q} D_Q(S) \tag{5.1}$$

such that

$$R \le R_{Budget} \tag{5.2}.$$

By introducing the Lagrangian cost function *J* and the corresponding Lagrange multiplier $\lambda \ge 0$ we can turn the above problem into a simpler, but not completely equivalent, unconstrained problem. The cost function becomes

$$J_Q(S) = D_Q(S) + \lambda R_Q(S) = \sum_{n_i \in S} [D_q(n_i) + \lambda R_q(n_i)] \tag{5.3}.$$

Minimizing equation (5.3) yields

$$\min_{S} \min_{Q} J_Q(S,\lambda) = J_{Q^*}(S^*,\lambda) \qquad (5.4).$$

The superscript * denotes that the optimum solution is found. It can be shown that *if and only if* the solution to (5.4) gives a rate equal to the rate budget for some fixed $\lambda \geq 0$, i.e. that $R_{Q^*}(S^*) = R_{Budget}$, then the unconstrained solution equals the constrained solution. So how do we interpret equation (5.4)? The equation expresses that by solving it, one obtains the optimal (in a rate-distortion sense) admissible WP-tree as well as the corresponding quantizer for each node. But this is for a fixed $\lambda \geq 0$. An extension is clearly needed so that one finds the optimal lambda as well.

For a given quantizer, one can calculate the corresponding rate and distortion. The set of quantizers and corresponding distortions for the solution to equation (5.4) is denoted an operating point $x = (R, D)$. Any operating point found with the unconstrained method will always be on the convex hull boundary of the rate-distortion curve. The convex hull [31] of the points $\{x_i\}_{i\in\{1,2...,N\}}$ is defined as the minimal convex set [30] containing these points. Figure 19 shows the relationship between the operating points and the convex hull. One can see that the operating points found with the unconstrained method define the vertexes on the convex hull boundary. A geometric interpretation to the solution of equation (5.4) is as follows: By moving a plane wave with slope -λ from origo towards the convex hull boundary in the R-D plane, the optimal operating point is the first point "hit" by this wave.

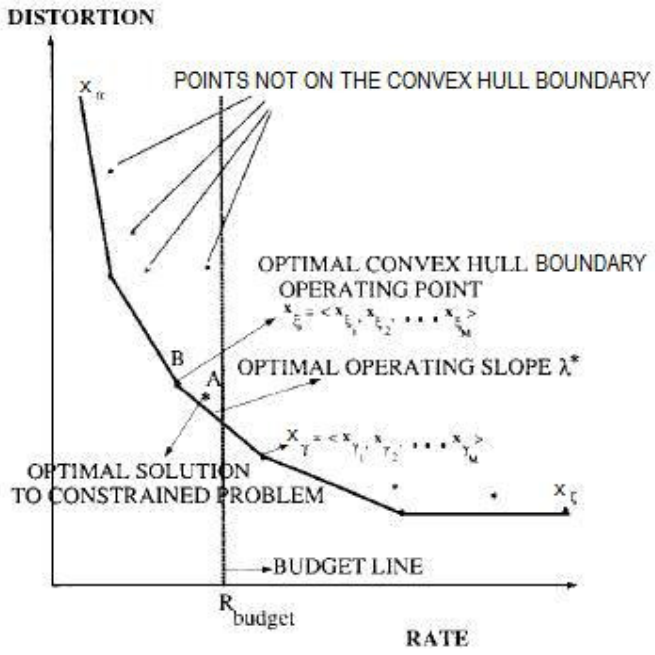


Figure 19 Relationship between rate-distiorion curve, operation points, convex hull, rate budget and lambda [29].

In order to find the optimal Lagrange multiplier λ^* from any initial value, equation (5.4) must, as previously mentioned, be extended somewhat. So let us introduce the biased Lagrangian cost functional where the rate budget is taken into account:

$$W(\lambda) = \min_S \min_Q [J_Q(S,\lambda)] - \lambda R_{Budget} = \min_S \min_Q [D_Q(S) + \lambda R_Q(S)] - \lambda R_{Budget} \quad (5.5).$$

It can be proven that $W(\lambda)$ is concave [30], and that it has one or two maximum values. The solution to

$$\max_\lambda W(\lambda) = \max_\lambda \left[\min_S \left[\min_Q [\sum_{n_i \in S} [D_q(n_i) + \lambda R_q(n_i)]] \right] - \lambda R_{Budget} \right] \quad (5.6)$$

indeed gives the optimal operating point on the convex hull boundary. If there are two maximum points one of them always corresponds to a rate above the rate budget while the other corresponds to a rate below the rate budget. Then one must choose the lower rate operating point.

Relating the solution of equation (5.6) to Figure 19 one can see that one finds the operating point B on the convex hull boundary which is closest to the rate budget. We will show that it is possible to find this point with a fast convex search algorithm. The constraint case, where one exhaustively searches through all possible combinations of operating points, also gives access to operating points not on the convex hull boundary. These may in some cases be more optimal than the points on the boundary itself, for which an example is illustrated with the point A, but this method is computationally too expensive.

Note that equation (5.6) can be interpreted as three independent optimization operations:

- The innermost minimization selects the quantizers for each node which minimizes $\sum_{n_i \in S} [D_q(n_i) + \lambda R_q(n_i)]$ with respect to the quantizers for a fixed λ.
- A next operation finds the optimal admissible tree.
- The outermost maximization determines the optimal $\lambda = \lambda^*$.

A pseudo code for this bit allocation scheme is presented in Appendix F.

5.4.1.1 Modifications to the algorithm

In this thesis, the same set of quantizers is chosen for each node. Thus $\mathbf{q}(n_i) = \mathbf{q}$ is independent on *n*. Further, one has the opportunity of, instead of calculating the real distortion for each quantizer, estimating it using equation (3.8). While estimation is much less computational expensive, the fidelity of the output signal will most likely decrease. But as we will see in the results section, the error introduced by estimation is of minor importance. As equation (3.8) is based on a standard uniform quantizer and not on a uniform quantizer with dead-zone which is used in our encoding scheme, an additional error will be introduced.

As discussed in section 5.2, the low-pass band is allocated zero bits. Since we claim that this does not introduce any distortion with respect to our application, the distortion associated with this node is also set to zero. This will clearly influence the choice of nodes in the WP-tree. It is also worth noting that even though this algorithm is designed for orthogonal wavelet filters, herein it is applied with a biorthogonal filter as well. Thus one introduces an error when calculating the distortion. The biorthogonal filter chosen is very close to be orthogonal though [24], so the error introduced is minor.

5.4.2 Greedy algorithm

This algorithm carries out a simple and intuitive idea, giving away bits to the subband needing it most at that moment. After reducing the subband's demand for more bits, this procedure is performed iteratively until one runs out of bits. The demand for bits is, as discussed in Appendix G, decided by the subband's variance. This algorithm does only take into account the maximum-depth-nodes of the WP-tree, which is equivalent to uniform subband coding. This algorithm is based on ideas described in e.g. [13],[32]. The algorithm is relatively simple and goes as follows:

1. Create the complete WP-tree with the algorithm described in section 3.3.3. Initialize $R_{used} = 0$.
2. Calculate the variance for each node at maximum tree depth except for the low-pass node. The variance for the low-pass node is set to zero due to the zero-allocation procedure discussed in 5.2.
3. Give a bit to the subband with the largest variance and reduce this subband's variance with a factor k. Also increase R_{used}; $R_{used} \leftarrow R_{used} + m$ where m is the number of coefficients in that subband.
4. If $R_{used} > R_{Budget}$, then stop. Else go to step 3.

It can be proven that for gaussian sources, $k = 4$ is the optimal factor with respect to the signal to noise ratio when using pdf-optimized quantizers [33]. The factor k is from here on denoted the *bit allocation parameter*, and it is optimized for our application in Appendix I.

5.5 Scalar uniform quantization with dead-zone

Scalar uniform quantization is a low complexity quantization scheme. A variant of the midtread quantizer discussed in section 3.2.1, denoted uniform quantizer with dead-zone, is the chosen quantizer in our encoding scheme. The difference from the midtread quantizer is that one extends the representation interval around zero, hence the name dead-zone. Mathematically the dead-zone is of size $k\Delta$ where k is a constant above 1 and Δ is defined as in equation (3.7). Let us denote k as the *dead-zone parameter*. The size k has influence on the number of coefficients represented with zero. A large dead-zone will remove many small coefficients while increasing the resolution of the larger values for a given rate budget. A figure of a dead-zone quantizer is shown below.

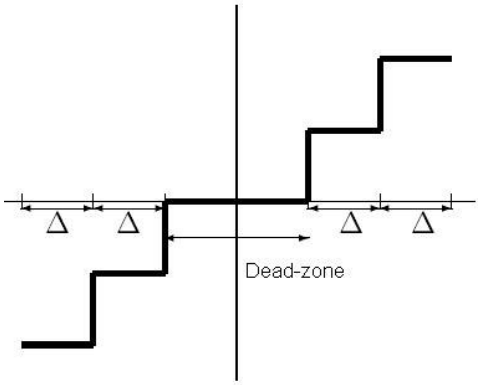


Figure 20 A dead-zone quantizer

The dead-zone quantizer is employed with success for image-coders, it is e.g. used in the JPEG2000 scheme [8]. It will definitely lead to longer zero-runs for our encoding scheme also, and together with run-length encoding the compression will increase compared to a standard midtread quantizer. This will be discussed further in the results section.

5.6 Run-length encoding

Run-length encoding is an efficient lossless source coding technique when there are long runs of one specific symbol. It is not optimal with respect to the source-entropy but has advantages such as low complexity and good performance in many practical situations. Run-length encoding is e.g. a part of the JPEG-standard [34].

The idea of run-length encoding is to replace a string of repeated symbols with a single symbol and a run-length indicating the number of times the symbol is repeated.

As an example, the string:

"*aaaabbcdeeeeefghhhij*"

may be replaced with:

"*a**3**b**1**cde**4**fgh**2**ij*"

by using a simple run-length scheme. But how do we know whether a symbol is being followed by another symbol or a run length? A solution to this is to repeat a symbol which repeats itself once before writing the run-length of the remaining symbols. Using this scheme, which is the chosen method for our implementation, the example string is encoded like this:

"*aa**2**bb**0**cdee**3**fghh**1**ij*"

After the run-length encoding is performed for one subband, the length of the coded coefficients is compared to the length of the original coefficients. If no compression is achieved, one sends of course the latter.

Note that the run-length encoding is not included in the bit-allocation schemes in order to maintain a low complexity. Thus one may obtain lower rate than the rate budget has specified.

5.7 Remarks regarding the decoder

The decoder performs in general the inverse operations of the encoder, i.e. run-length decoding, inverse quantization and signal reconstruction. When it comes to reconstruction, recall that the time resolution of the preprocessed signal is halved compared to the original input signal in the encoder. The decoder only reconstructs the signal to the same time resolution as the preprocessed signal. The last interpolation step is performed in MATLAB with the function "*interp*." The MATLAB interpolation filter is an optimum filter in the mse-sense [35]. Hence one increases the fidelity of the output compared to using a filter with lower complexity.

6 Results and discussion

This section presents the results and the related discussion both regarding the processing time and the optimization/performance of the encoding scheme. A fidelity measure for the compressed traces is also defined herein. Note that many additional results are shown in Appendix I.

6.1 Processing time vs. transfer time

When transferring data, an ultrasonic trace is supposed to be encoded while another one is being sent. Assume that the processing time for a specific encoding scheme is constant regardless of the target bitrate. First we consider the task of minimizing the transfer without so much emphasis on the fidelity. Then we claim that in an optimal system with respect to transfer time, this merit equals the processing time. To see this, consider the following: If the processing time exceeds the transfer time, the transmitter has to wait for the processing unit to finish. A slight decrease in the compression ratio will clearly increase the transfer time for each trace. But as long as the transfer time is less than the processing time, there is no increase in the overall time of communication while the fidelity of the sent signal is better. On the other hand, if the transfer time exceeds the processing time, the encoded signal must be put in a send-queue. In this case it would be more optimal to have a higher compression ratio, thus lowering the transfer-rate, until the two merits are equal.

The MicroBlaze microcontroller has a built-in function for counting the number of clock-cycles elapsed during processing. With an operating speed of 50 MHz, the processing time, t_p, can be found as follows:

$$t_p = \frac{number_{clock\_cycles}}{50MHz} \tag{6.1}.$$

Having the opportunity to use two different filters and two different encoders, give rise to four different graphs of processing times vs. bitrate. This is illustrated in Figure 21. Optimality in the abovementioned sense is fulfilled when the graphs of the processing-times crosses the graph of the transfer-time. This is denoted an *operating point* (do not mix these with the operating points discussed in section 5.4.1.)

Note that the distortion is estimated (and not calculated) for the BWP-cases above. To get an idea of the processing times when calculating the real distortions, see Table 1. The transfer time, t_t, in milliseconds is a linear function of the total rate, and with a bitrate of 115200 bits/second the mathematical relation is as follows:

$$t_t = \frac{580 + 1.25 * total\_rate}{115.2} \tag{6.2}.$$

The constant term, 580, is due to header data which is independent of the encoder, *total_rate* equals bitrate times 2048 and the factor 1.25 is for administrating the link. Note that the assumption of a constant processing time regardless of the total rate is not far from true. The increase is mostly due to more binary coding. The small fluctuations seen when using the 10/10-filter together with the BWP-algorithm is expected due to the iterative approach for finding the optimal admissible tree. One

would probably get a similar result for the 7/9-filter with more data-points. Such fluctuations will not occur for the greedy-algorithm, due to its deterministic nature.

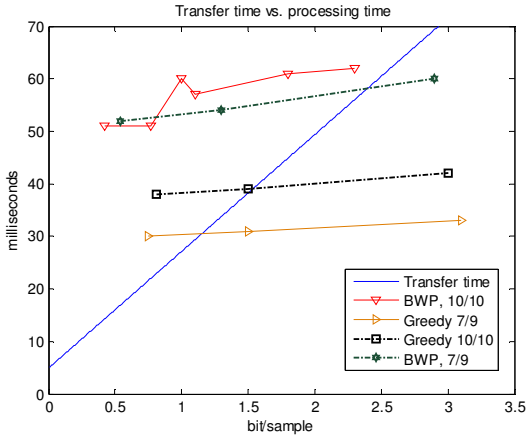


Figure 21 Transfer time vs. processing times

The cost of the filtering process compared to the total encoding costs is presented in Table 1.

Table 1 Total processing times and the cost of filtering for the two algorithms and filters. The rate budget is 3500 bits in each scenario. The values in the paranthesis are the cost of filtering compared to the total encoding costs. To the right is two BWP-cases where the real distortion is calculated. The bitrate corresponding to a rate budget of 3500 bits is also presented.

	BWP estimated, 10/10	BWP estimated, 7/9	Greedy, 10/10	Greedy 7/9	BWP, 10/10	BWP, 7/9
Total	60 ms	54ms	39 ms	31 ms	127 ms	114 ms
Filtrering	33 ms (55%)	25 ms (46%)	34 ms (87%)	26 ms (84%)	33 ms (26%)	25 ms (22%)
Bitrate (bit/sample)	1.0	1.3	1.5	1.5	1.1	1.3

If we try to summarize, the following conclusions can be drawn:

- Filtering with the 7/9-filter saves about 25% of the processing time compared to the 10/10-filter. This is both due to fewer taps and thus fewer multiplications, and due to the non-increasing number of coefficients.
- The greedy algorithm saves 35% - 45% of the processing time compared to the BWP–algorithm when estimating the distortion.
- It is not of practical interest to use the BWP-algorithm without estimating the distortion due to the high processing-time.
- The operating point corresponding to the lowest bitrate (1.2 bit/sample) is achieved by using the greedy algorithm together with the 7/9-filter. This is from here on referred to as *the optimal operating point* for our encoding scheme.
- Due to the iterative approach for finding the optimal solution using the BWP-algorithm, the processing times may fluctuate. Concavity of the cost function ensures convergence, but the number of iterations may differ from trace to trace.
- If we compare the operating points for the greedy algorithm with the transfer time of an unprocessed trace (see section 2.4), the time of transmission is 9

and 12 times faster after compression when using the 10/10- and 7/9-filter respectively. The BWP-algorithm reduces the transfer time with a factor of 6.

6.2 Fidelity measure

It is not enough only to take into account processing- and transfer-time as we did in the last section. The most important is off course that the decoded signals have the desired fidelity. Including fidelity as an additional merit of optimization, *the optimal system is therefore a system which, simultaneously, has traces with the desired fidelity criterion as its output, where the processing time equals the transfer time and where the transfer time is minimal.*

Note that we in the above definition indirectly have incorporated the compression ratio: A minimal transfer time yields a minimal processing time which in turn implies a high compression ratio.

For audiovisual source coding, the fidelity criteria are well defined. Regarding ultrasonic traces applied for thickness measurements, on the other hand, no such criteria exist. The ultrasonic traces in this thesis are meant for absolute thickness measurements of subsea pipelines. If several measurements are performed within a time-interval, it is important that one can make an educated guess of the long-term corrosion rate.

SensorLink AS has developed two models for measuring the thickness based on a given ultrasonic trace:

- A model-based inversion method
- A method based on correlation

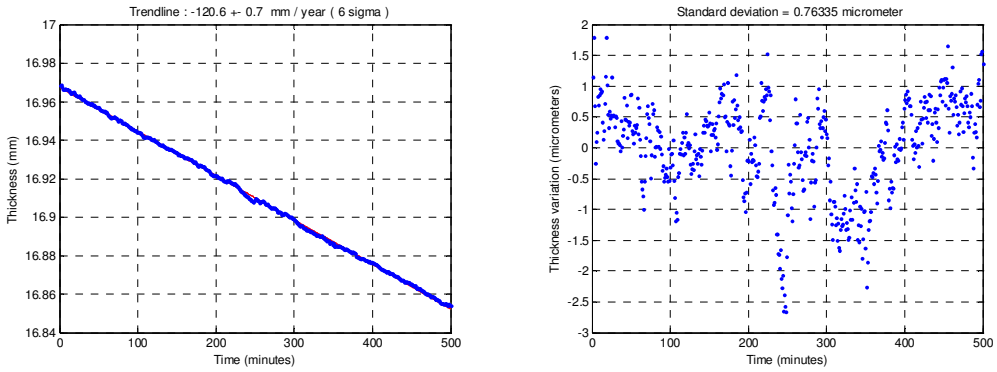
In order to see if the compressed traces have the desired fidelity with respect to the trendline, the following procedure is performed:

1. First, 500 consecutive traces from both the low-rate and the high-rate corrosion experiments discussed in section 4.1 are selected. Thus we have two sets of traces. The sets are chosen in time-intervals where the corrosion rates are relatively linear. Thus the corrosion rate can be approximated by a linear trend.
2. Calculate the thicknesses and find the corrosion-rate from the uncompressed traces by using the models mentioned above. Also calculate the standard deviation from the trend.
3. Compress the traces and do the same calculations once more.
4. Compare the results of the compressed and uncompressed traces.

A variation to the above procedure is to compare the measured absolute thicknesses for the compressed vs. uncompressed traces instead of focusing on the corrosion rate. This approach is taken in section 6.7.

The corrosion rate and standard deviation calculated with the inversion method from the 500 original high-rate traces are shown in Figure 22. The uncertainty of the calculated trendline (here +-0.7 mm/year) is related to the standard deviation; increasing the uncertainty also increases the standard deviation and vica verca. As the

measurements are taken each minute, 500 consecutive traces corresponds to 500 minutes.



a) b)

Figure 22 Trend-line (red) vs. actual thickness calculations (blue) in a), and standard deviation from the trend-line in b). It is almost impossible to see the trendline due to the good match with the measured data.

There are a variety of encoding-parameters which must be adjusted for optimization purposes. In order to limit the number of test we have to be selective. Moreover, it is often assumed that some the optimization parameters are independent. Thus one can e.g. optimize the dead-zone for the orthogonal filter, and assume that this is optimal for the biorthogonal filter too. Such an assumption, although not completely correct, will reduce the number of tests drastically.

The following parameters are assumed throughout this chapter:

- The bit allocation parameter equals 4.
- The inversion method is used for thickness calculations.

The justification of these choices is based on optimizations performed in Appendix I.

6.3 Explanation of results

Trendlines from the compressed high-rate and low-rate traces are always compared with the original trendline corresponding to the unprocessed traces in Figure 22 a) and Figure 26 a) respectively. A typical trendline resulting from the compressed high-rate traces is presented in Figure 23.

Examining Figure 22 a) and Figure 23 the trendlines have the numerical values $-120.6mm/year \pm 0.7mm$ and $-124mm/year \pm 3mm$ respectively. Now three terms are introduced:

- Deviation from original trendline
- Deviation from original uncertainty
- Relative uncertainty

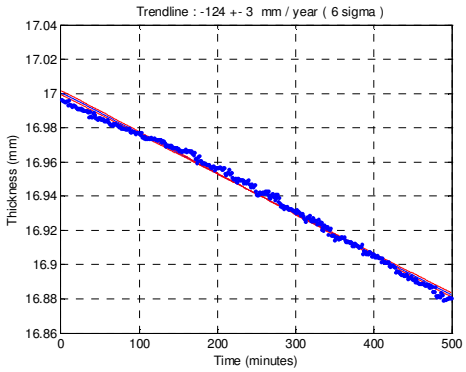


Figure 23 Typical trendline from compressed high-rate traces. In this case the bitrate is 1 bit/sample

The deviation from the original trendline is measured in absolute value, and in this example it is $|-120.6-(-124)|mm/year = 3.4mm/year$. This term is thus actually related to the slope of the trendline. The deviation from the original uncertainty is accordingly $(3-0.7)mm/year = 2.3mm/year$. We want of course both of these quantities to be as low as possible, but generally more emphasis is put on the deviation from the original uncertainty. This is because a small deviation from the original uncertainty excludes the possibility of having a large deviation from the original trendline, while a small deviation from the original trendline *does not* exclude the possibility of having a large deviation from the original uncertainty.

The relative uncertainty is defined as the ratio

$$rel\_unc = \frac{uncertainty}{slope\_of\_trendline} \qquad (6.3).$$

For the trendline in Figure 23 the relative uncertainty thus becomes 3/124 = 2.4%. The steeper the slope, the larger uncertainty one can tolerate for a given relative uncertainty.

If we now vary one parameter (e.g. the dead-zone or rate budget) while keeping the others constant, we will obtain multiple such trendlines with their corresponding deviations from the original trendline and original uncertainty. Thus one can create graphs, for which some are shown in Figure 31 and Figure 32, of how the deviations vary with a given parameter.

For the compressed high-rate traces we define the fidelity as acceptable as long as the relative uncertainty is below 2%. If the relative uncertainty exceeds this, the unwanted "stair-case"-effect explored in Appendix I comes into play. For the compressed low-rate traces we define the fidelity as acceptable as long as the relative uncertainty is below 7.3%. This corresponds to an uncertainty of about +-0.1 mm/year which is the system requirement for this kind of experiments.

6.4 Preprocessing and zero-allocation

This section discusses the effects of the preprocessing as well as the zero-allocation of the low-pass band. Hence the following is performed on each trace:

- Identify the first reflection and cut the signal 300 samples in front of that.
- Decimate with a factor 2 using the Haar filter. The corresponding filter coefficients are presented in Appendix B.
- Decompose the signal into a level 4 WP-tree. Remove the lowest frequency-band of level 4 and reconstruct the signal.

A visual inspection of an original trace compared to the reconstructed trace after preprocessing and zero-allocation yields:

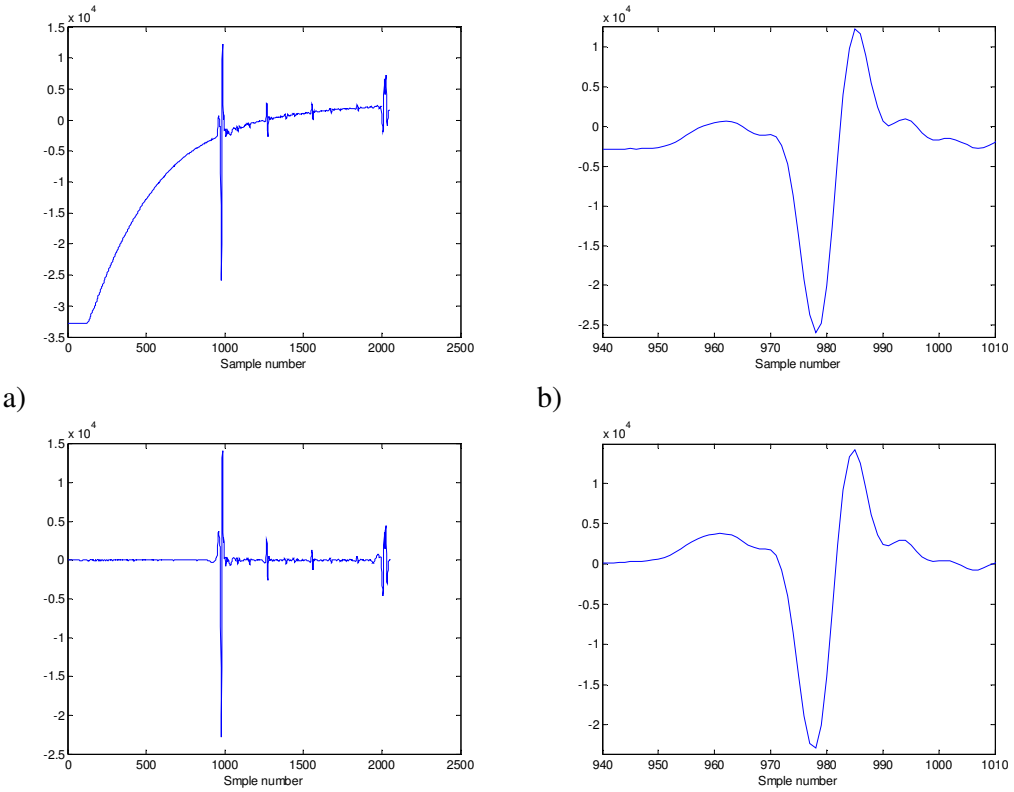


Figure 24 The original signal and zoom of the according first reflection in a) and b) respectively. The reconstructed signal after preprocessing/zero-allocation and zoom of the according first reflection in c) and d) respectively.

When comparing Figure 24 b) and d), it is difficult to see any difference in the pulse-form.

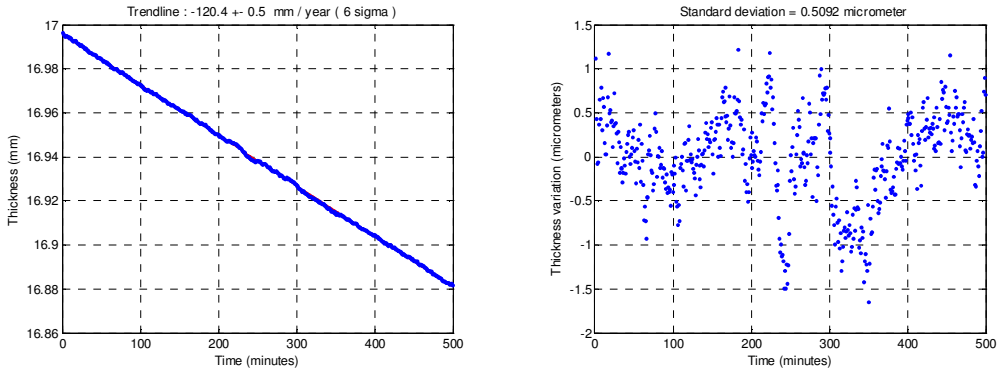
6.4.1 High-rate corrosion

While the 10/10-filter is applied for the results in this section, one obtains similar results with the 7/9-filter. See Appendix I for additional results regarding the correlation method and 7/9-filter.

6.4.1.1 High-rate experiment with preprocessing and zero-allocation

The trendline and standard deviation corresponding to the preprocessed and zero-allocated signal is presented below. The following parameter is chosen:

- Filter type: 10/10



a) b)

Figure 25 Corrosion rate in a), and standard deviation from the trend-line in b).

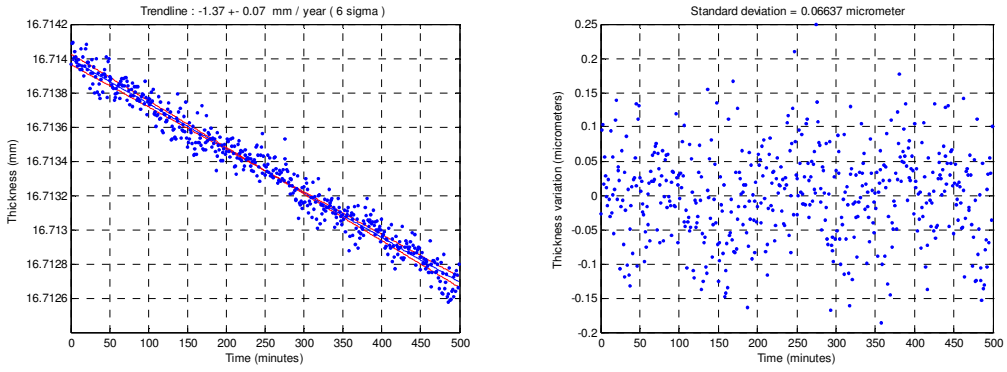
The trendline from the reconstructed traces in Figure 25 is about equal to the trend-line in Figure 22 while the standard deviation actually has decreased. The fact that the noise in Figure 25 a) and Figure 22 a) do not appear to be quite white suggest that there are small variations in the induced current during the experiment, leading to small variations in the corrosion rate. But the variations are so small compared to the corrosion rate that one cannot notice them when looking at the trendlines.

6.4.2 Low-rate corrosion

In this experiment the corrosion rate is only about a hundredth of the high-corrosion rate. A lower corrosion rate leads to less difference between to consecutive traces. We can therefore expect a larger sensitivity to the preprocessing and zero allocation. This results in a greater need for optimization than in the high rate case.

6.4.2.1 Original traces

The trendline and standard deviation for the unprocessed traces are presented below. Note that the noise is much "whiter" here than in the high rate corrosion experiment. This indicates that no systematic errors, such as temperature variations or induced current variations, are introduced during the experiment.



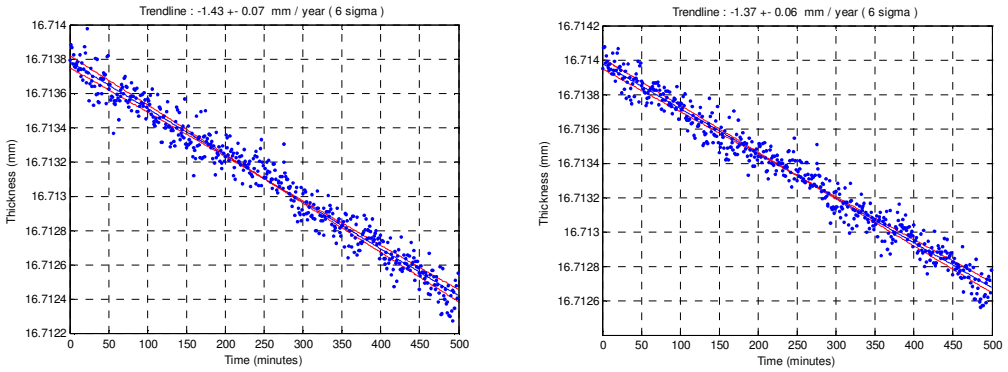
a) b)

Figure 26 Trendline in a) and corresponding standard deviation in b).

All results regarding the low rate corrosion experiment will hereafter be compared with the trendline in Figure 26. Note that the uncertainty for the trendline is +-0.07 mm/year, while it in the high rate case is about 10 times larger. But relatively, compared to the slope of the trendline, it is actually the uncertainty of the low rate case which is 10 times larger than that of the high rate case. To see this, consider the relative uncertainty which is 0.07/1.37 = 5% for the low rate case and 0.6/120.6 = 0.5% for the high rate case. This indicates a factor of 10. Note also from Figure 26 b) that the standard deviation from the trendline is only 66 nm.

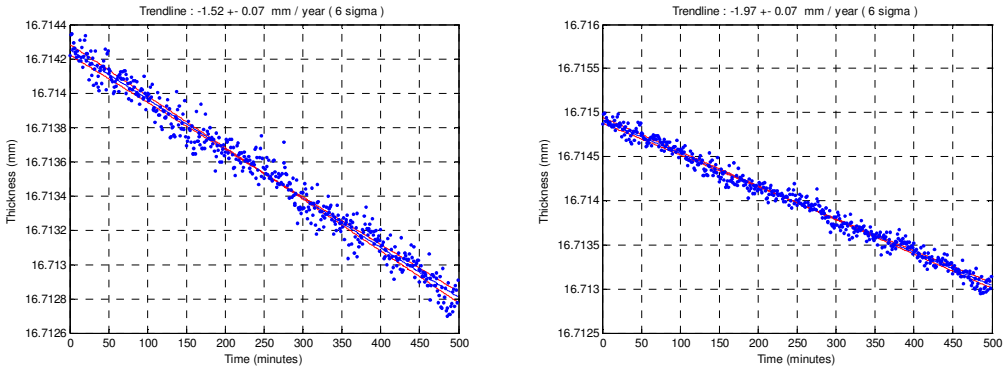
6.4.2.2 Preprocessing and zero-allocation

In this section we discuss whether preprocessing or zero allocation introduces any distortion. The preprocessing with the Haar filter is also compared with a corresponding procedure where the MATLAB built-in function "*decimate*" is used instead.



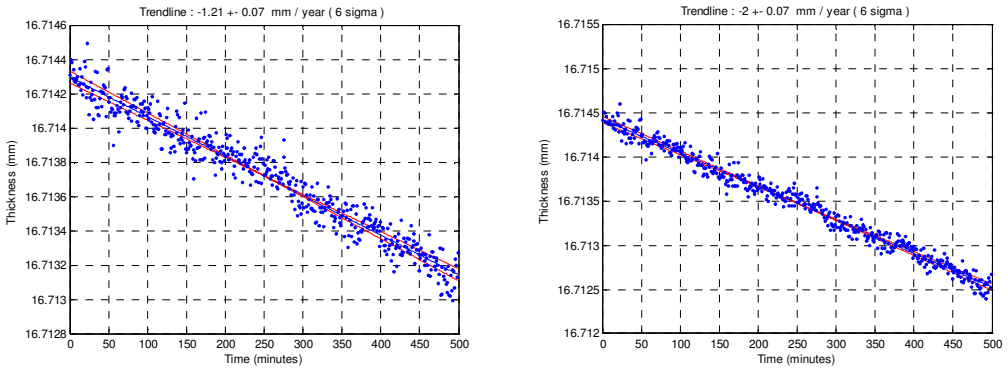
a) b)

Figure 27 Preprocessing only. Haar filter is used in a) and the MATLAB-function "*decimate*" is used in b).



a) b)

Figure 28 Zero allocation only. The 10/10 filter is used in a), while the 7/9 filter is used in b).



a) b)

Figure 29 Both preprocessing with the Haar-filter and zero allocation. The 10/10 filter is used in a), while the 7/9 filter is used in b).

By examining Figure 27 and comparing with the trendline from the original traces in Figure 26, we see that decimating with the Haar filter introduces some distortion while using the MATLAB-function "*decimate*" does not. The distortion introduced when using the Haar filter is acceptable, but should still be reduced. There are other low-complexity filters which should improve on this while maintaining a low processing time. Note also, by comparing Figure 27 b) and Figure 26 a), that the uncertainty has decreased from +-0.7 mm/year to +-0.6 mm/year when using the "*decimate*"-function. This is probably due to removal of high-frequent noise. Hence a LP-filter should be considered in an updated model for calculating the thickness.

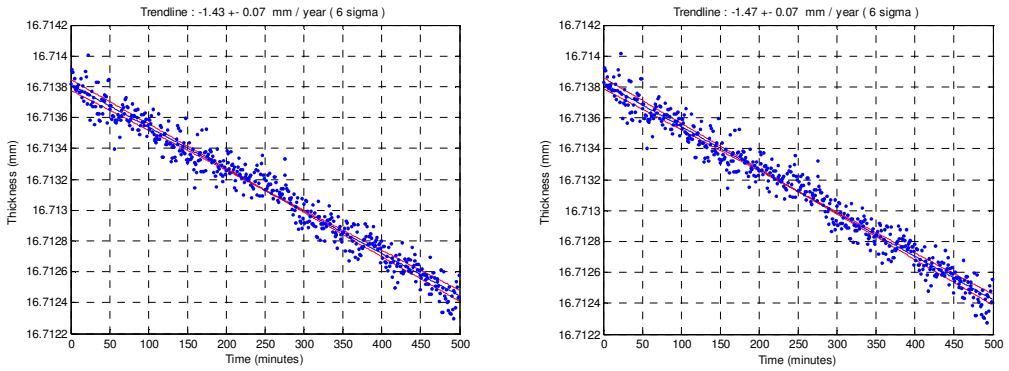
The zero allocation introduces more distortion than the preprocessing, a fact which is seen by comparing Figure 28 and Figure 27. Note also from Figure 28 that the 10/10-filter performs better than the 7/9-filter, with deviations from the original trendlines of 0.15 mm/year and 0.60 mm/year respectively.

Combining preprocessing and zero allocation yield the results seen in Figure 29. We see that the preprocessing does not introduce much more distortion from the original trendline than already introduced by zero allocation.

Note that in all these results, the uncertainty remains constant except for the "*decimate*" case where the uncertainty has decreased. It is the slope of the trendline which varies.

Improvement of the zero-allocation scheme

Since the zero-allocation clearly introduces some distortion, a suggestion for improvement is to decompose the LP-node one more level before zero allocation. Note that the level 4 LP-node consists of max 50 coefficients. The cost for one more level of decomposition for this node only, is minimal. The corresponding results are presented in the figure below. Note that this is only performed in MATLAB, and not implemented into the C-code and tested together with compression. That is considered as future work.



a) b)

Figure 30 Both preprocessing with the Haar-filter and zero allocation. Now the LP-node is decomposed to level 5. The 10/10 filter is used in a), while the 7/9 filter is used in b).

Comparing Figure 30 with Figure 27 a), we can see that the zero allocation does not introduce more error than already introduced by the preprocessing when using the 10/10-filter. The 7/9-filter, on the other hand, still introduces some additional distortion. But anyway, there is a huge improvement compared to the results shown in Figure 29.

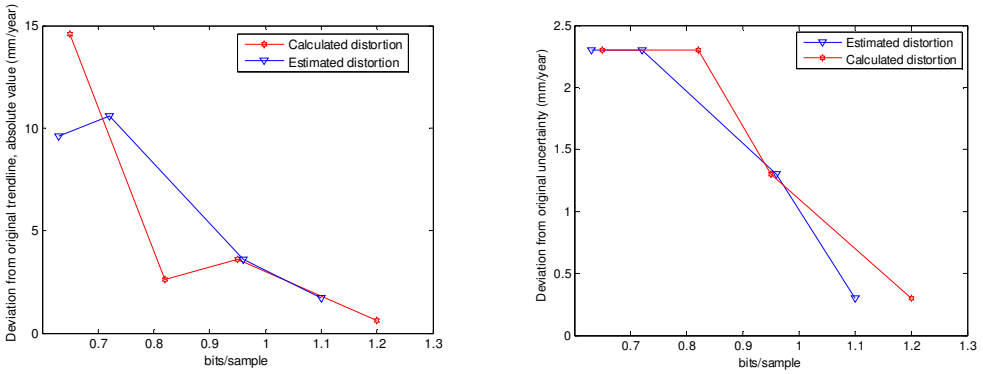
6.5 Compressed traces – high-rate corrosion experiment

This section deals with the results regarding the compression of traces from the high-rate corrosion experiment. The results are mostly related to optimization of different parameters. Note that if not otherwise stated, the distortion is always estimated when employing the BWP-algorithm.

6.5.1 Calculated vs. estimated distortion for the BWP-algorithm

This is a test for verifying the robustness of the distortion-estimation for the BWP-algorithm by comparing it with the results obtained when calculating the real distortion. All the parameters but the rate budget are held constant. The following parameters are chosen:

- Filter type: 10/10
- Dead-zone parameter: 3



a) b)

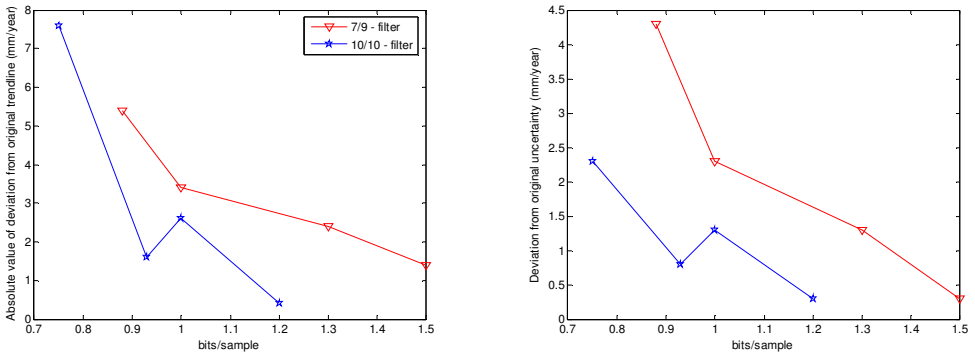
Figure 31 Deviation from original trendline in a) and deviation from original uncertainty in b) with respect to bitrate.

The result for the unprocessed traces is shown in Figure 22. The deviations from the original trendline and original uncertainty generally are as expected decreasing with the bitrate, as seen in Figure 31 a) and b). There are small differences in the results between calculating and estimating the distortion, and we can thus conclude that the performance is not significantly degraded by using the latter scheme. For low bitrates, when the deviation from the original uncertainty exceeds two, a phenomenon denoted the *staircase effect* occurs. This is examined in more detail in Appendix I.

6.5.2 Biorthogonal vs. orthogonal wavelet filter

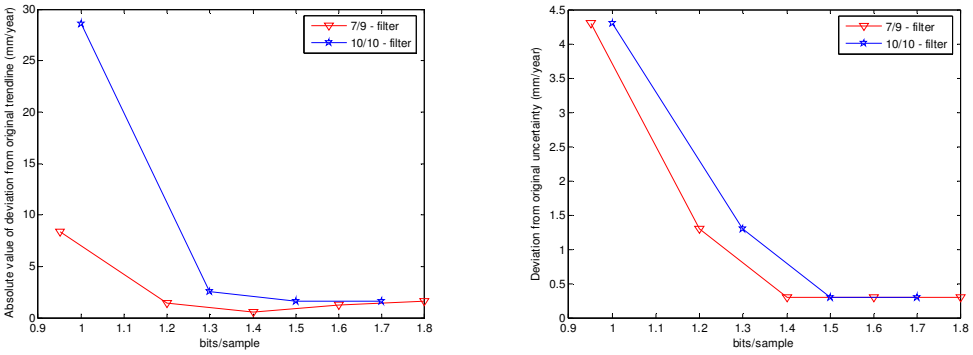
These simulations compare the 7/9-filter vs. the 10/10-filter. The following parameter is chosen:

- Dead-zone parameter: 2.5



a) b)

Figure 32 Deviation from original trendline in a) and deviation from original uncertainty in b) with respect to bitrate. In this scenario the BWP-algorithm is applied for bit allocation.



a) b)

Figure 33 Deviation from original trendline in a) and deviation from original uncertainty in b) with respect to bitrate. In this scenario the greedy-algorithm is applied for bit allocation.

The result for the unprocessed traces is shown in Figure 22. The deviations from the original trendline and original uncertainty are as expected generally decreasing with the bitrate. Examining Figure 32 and Figure 33 more carefully, also reveals the peculiar result that, for a given bitrate, the 10/10-filter seems to perform better than

the 7/9-filter when using the BWP algorithm, while the opposite is the case with the greedy-algorithm.

The abovementioned results are partly related to the run-length encoding, but how? Because of coefficient expansion when using the 10/10-filter, the BWP algorithm tends to choose WP-nodes which are not on the maximum tree-depth. Recall that our WP-tree has 4 levels. With the 7/9-filter, the BWP algorithm chooses more nodes on level 4 due to zero coefficient expansion. We can assume that the largest coefficients in a subband correspond to one of the ultrasonic reflections. Further, the time resolution decreases with tree-depth. Thus the number of samples between the reflections is reduced by a factor 2 for each new level one moves toward the maximum tree-depth. Evidently it is the samples in between the reflections which will turn into zero-runs after quantization. The conclusion is thus that using the 10/10-filter leads to more low-level nodes which in turn give rise to longer zero-runs. The longer the zero-runs the more compression is performed by the run-length encoder. The same conclusion can be drawn when comparing the greedy algorithm vs. the BWP algorithm as the greedy algorithm only chooses the maximum level nodes. Thus the run-length encoder will be more efficient together with the BWP algorithm.

The reason why the greedy algorithm performs better with the 7/9-filter cannot be explained with the run-length encoder since only the level 4 nodes are used. One obvious explanation is the coefficient expansion when using the 10/10-filter; while the level 4 nodes have 41 coefficients each without expansion, it has 49 when expansion occurs. This gives a total of 120 extra coefficients which corresponds to a 20% increase (735 instead of 615 coefficients when disregarding the LP-node which is removed). It is clear that such an increase of coefficients will affect the compression results.

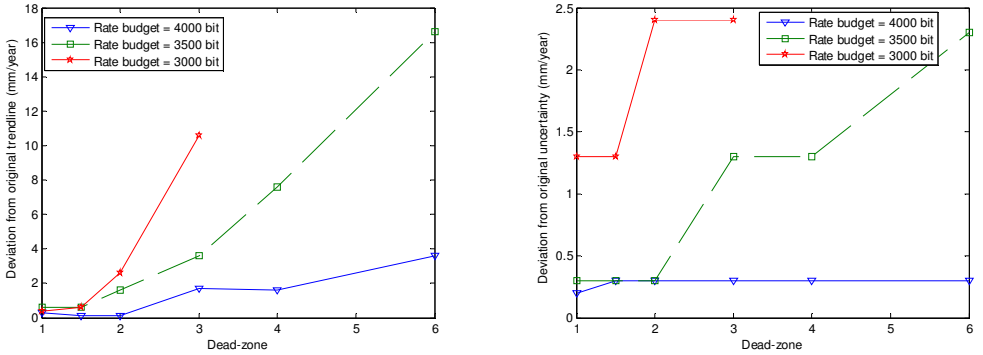
These results should have an impact on future design of the encoder:

- As the run-length encoder works better for the low-level nodes, it is not necessary to create a full WP-tree. Maybe dyadic wavelet decomposition suffices. This will reduce the processing time, generate better compression as long as only the simple run-length encoding scheme is employed, and one may still use the encoding schemes described in section 5 after small modifications.
- Incorporating the run-length encoder into the BWP-algorithm will improve the algorithm significantly since it will consider the nodes optimal with respect to run-length encoding in a completely new way. The downside with this approach is an increase in the processing time.

6.5.3 Optimization of the dead-zone parameter

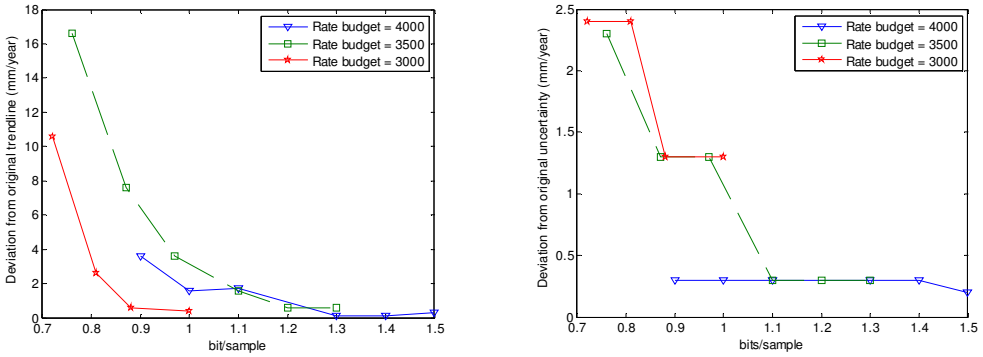
In this section several plots are presented in order to understand how different dead-zone-sizes affect the compression ratio and fidelity of the output. In each figure there are three graphs, each representing a specific rate budget. Recall that as the run-length encoding is not included in the bit-allocation scheme, the size of the output is generally lower than the rate budget suggests. The following parameters are chosen:

- Bit allocation scheme: BWP-algorithm
- Filter type: 10/10



a) b)

Figure 34 Deviation from original trendline in a) and deviation from original uncertainty in b) with respect to the dead-zone parameter.



a) b)

Figure 35 Deviation from original trendline in a) and deviation from original uncertainty in b) with respect to bitrate.

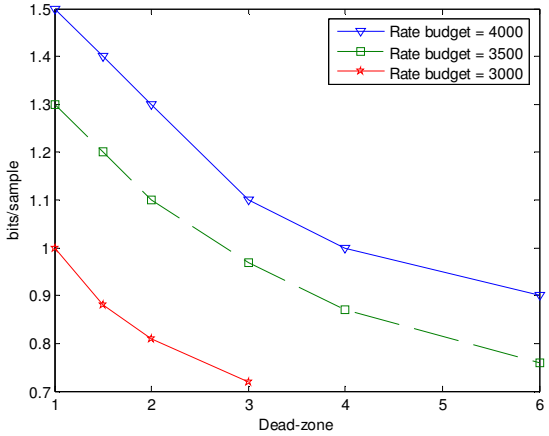


Figure 36 Bits/sample with respect to the dead-zone parameter. It is clear that a larger dead-zone parameter increases the compression ratio.

Figure 34 demonstrates that the fidelity decreases with the dead-zone-size, especially for the two lower rate budgets. Figure 35 relates the fidelity to the bitrate instead of the dead-zone. By combining Figure 34 and Figure 35 one obtains Figure 36 which gives a direct relation between bitrate and the dead-zone. Further one can conclude from the figures that:

- A rate budget of 4000 bits yields good results for all the tested dead-zone parameters. The deviation from original uncertainty does not exceed 0.3 mm/year. With a dead-zone parameter of size 6, a bitrate of 0.9 bit/sample is achieved. The deviation from the original trendline is then 3.6 mm/year, corresponding to 3.6/120.6 = 3%.
- If one wants better compression than 0.9 bit/sample, a lower rate budget than 4000 bits must be used. But we see that the fidelity decreases fast below 0.9 bit/sample.
- It is not always clear if one should lower the rate budget or increase the dead-zone parameter. E.g. for a target bitrate of 1.1-1.3 bit/sample, one gets similar fidelity results with a rate budget of 3500 bits and 4000 bits by adjusting the dead-zone parameter, while for a target bitrate of 0.9 bit/sample one should either choose a rate budget of 3000 bits together with a dead-zone parameter of 1.5 or a rate budget of 4000 together with a dead-zone parameter of 6. One should therefore optimize this with respect to each final target bitrate.

One can clearly see that a larger dead-zone parameter affects the lower rate budgets more than it does when employing 4000 bits. The solution is simple: For a high rate budget, more bits are assigned to each subband. If e.g. a subband has the dynamic range [-128,128] and it is represented with 7 vs. 5 bits in the high rate budget- and low rate budget case, then the quantization levels are 2 and 8 respectively. An increase in the dead-zone parameter will then have a much larger impact on the low rate budget case because it is proportional to the quantization level. By increasing the dead-zone parameter with one, the dead-zone thus only increases with 2 in the former case while it increases with 8 in the latter.

6.5.4 *Optimality with respect to transfer- and processing time and the corresponding fidelity*

Until now we have discussed which impact variations of different parameters have on the fidelity of the output. But we have not related this to the discussion in section 6.1 about processing- vs. transfer time. Section 6.5.3 demonstrates that one can obtain satisfactory output for bitrates down to 0.9 bit/sample when using the BWP-algorithm together with the 10/10-filter. Relating this to Figure 21 we see that with the BWP-algorithm, such a compression ratio is unnecessarily high: It's no use to represent the source with less than 2.4 bit/sample because the processing time for that bit allocation scheme is too high. Thus we see that it is no problem to obtain acceptable fidelity at the corresponding operating points.

Using the greedy-algorithm and the 7/9-filter, on the other hand, one obtains the operating point with a bitrate of 1.2 bit/sample (recall that this corresponds to the optimal operating point for our encoding scheme.) Such bitrates are achieved by e.g. choosing the following parameters:

- Filter type: 7/9
- Rate budget: 3100 bits

- Dead-zone: 3.5
- Bit allocation scheme: Greedy

The resulting trendline as well as a comparison of the compressed waveform vs. the original waveform is shown in the figures below. We see in Figure 37 that the fidelity of the compressed data is acceptable, with a deviation from the original trendline of 4.4 mm/year, a deviation from the original uncertainty of 1.3 mm/year and a relative uncertainty of 1.6%. As this choice of parameters (greedy-algorithm and 7/9-filter) is the most critical with respect to bitrate, we can conclude that we achieve the desired fidelity for all operating points.

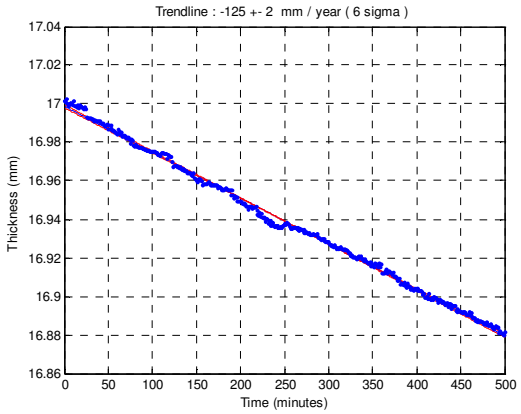
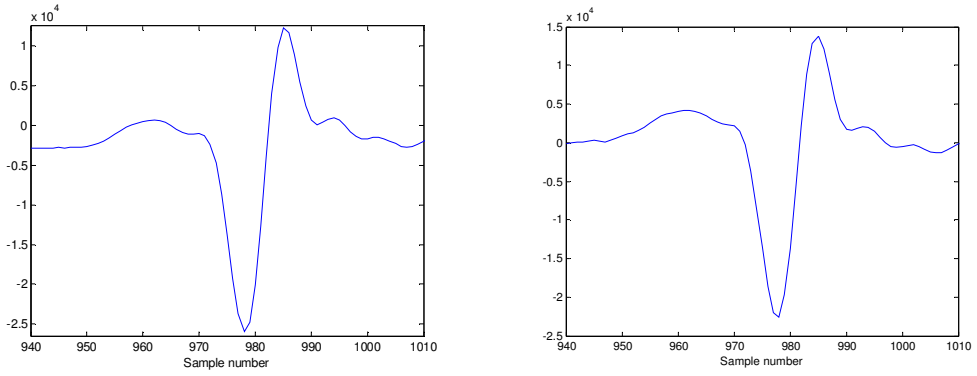


Figure 37 Trendline for compressed traces using the greedy-algorithm together with the 7/9-filter. The compression ratio is optimal with respect to processing- and transfer time, and we see that the fidelity is good.



a) b)

Figure 38 A visual inspection of original waveform in a) vs. the compressed waveform in b) shows that the distortion is small. The bitrate is 1.2 bit/sample; a compression ratio of 13.3.

6.6 Compressed traces – low-rate corrosion experiment

This section deals with the results regarding the compression of traces from the low-rate corrosion experiment. The results found in section 6.5 are assumed to apply here as well. Note that if not otherwise stated, the distortion is always estimated when employing the BWP-algorithm.

There is a factor of about 100 between the low corrosion rate and high corrosion rate. Thus the average change for 5 consecutive high-rate traces corresponds to the total

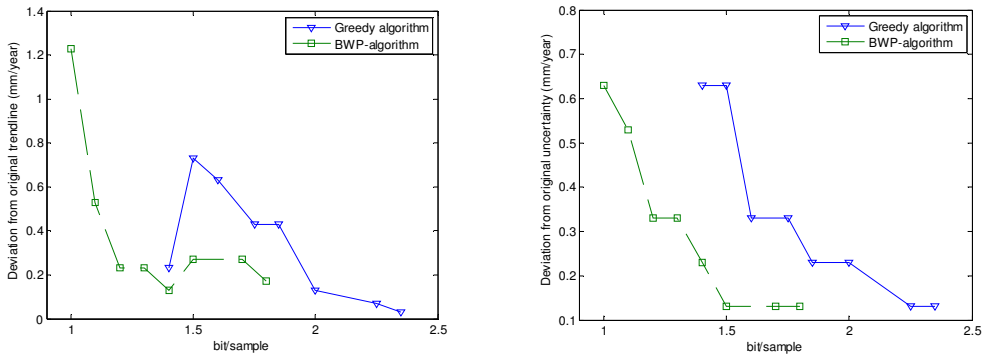
change for all 500 low-rate traces. Another point of view is this: While the average change in thickness corresponding to two consecutive traces is about 260 nanometer (nm) in the high-rate case, it is about 2.6 nm in the low-rate case. The latter corresponds to the diameter of a medium sized molecule! The total change in thickness corresponding to all the 500 low-rate traces is only 1.3 μm. It is evident that this set of traces will be extremely sensitive to lossy compression.

In this section the emphasis is not put so much on the compression ratio (or equivalently, how many bits/sample which is used.) First and foremost we are aiming at obtaining acceptable results at all, and we will not do a systematic optimization of the parameters as in section 6.5.

6.6.1 *Fidelity vs. bitrate*

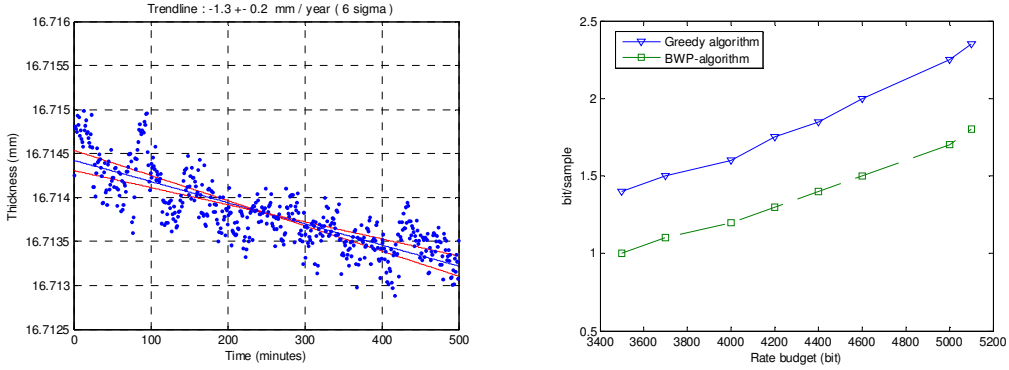
In this section we will discuss how the fidelity of the output of both bit allocation algorithms changes with the bitrate. This will also be related to the rate budget to see the performance of the run-length encoder. It was concluded in 6.4.2.2 that the 10/10-filter performs best in the low-rate cases. Thus the following parameters are used.

- Filter type: 10/10
- Dead-zone parameter: 2



a) b)

Figure 39 Deviation from original trendline in a) and deviation from original uncertainty in b) with respect to the bitrate.



a) b)

Figure 40 The trendline corresponding to one of the better test-cases in a) and bitrate vs. rate budget in b).

Figure 39 b) clearly demonstrates how the fidelity increases with the bit-rate. The fluctuations seen in Figure 39 a) is expected when the uncertainty is so high. Recall the general principle that a large deviation from the original uncertainty *does not* exclude a small deviation from the original trendline, while a large deviation from the original trendline *does* exclude a small deviation from the original uncertainty.

It is also clear that the BWP-algorithm performs better than the greedy-algorithm when it comes to bitrate. This is mostly related to the run-length encoding. See section 6.5.2 for a discussion about this theme. The relation between a given rate budget and the final bitrate is shown in Figure 40 b). Figure 40 a) displays the trendline when employing the greedy-algorithm and a rate budget of 5000 bits. We can see that the relative uncertainty equals 0.2/1.3 = 15%, which is a triplication compared to that for the unprocessed traces. This value is too high compared to the accepted fidelity-measure.

6.7 Absolute thickness measurements

Until now only deviations from the original trendlines and the corresponding uncertainty have been considered as fidelity measures. Section 6.2 also suggests that the absolute thickness may be used as an alternative fidelity measure. The same test procedure is implemented, i.e. comparing 500 unprocessed vs. compressed traces, but now the emphasis is put on the deviation from the absolute thickness of the unprocessed traces instead.

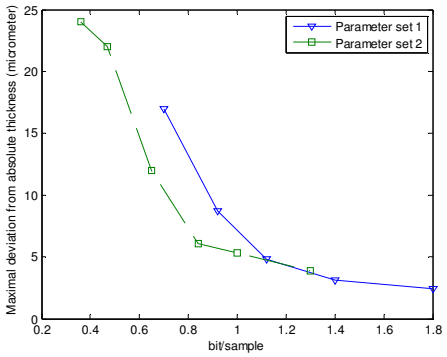
Recall that SensorLink aims at an uncertainty of +-0.1 mm for absolute thickness measurements. In what follows, plots of the maximum- as well as the average deviation between the thicknesses for the compressed vs. unprocessed traces is presented. For these tests, it's irrelevant whether the low- or high-rate traces are employed. Two parameter sets are chosen:

Parameter set 1:

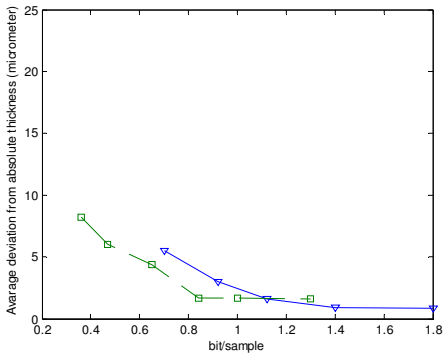
- Filter type: 7/9
- Corrosion type: High-rate
- Dead-zone parameter: 3
- Bit allocation algorithm: Greedy

Parameter set 2:

- Filter type: 10/10
- Corrosion type: High-rate
- Dead-zone parameter: 3
- Bit allocation algorithm: BWP



a)



b)

Figure 41 Maximal- a) and avarage b) deviation from the absolute thickness of the unprocessed traces.

For bitrates of about 1.1 bit/sample, the maximum deviation is 5 μm and the average deviation is about 1.6 μm. For bitrates below 1.1 bit/sample the BWP-algorithm performs better, which was pretty much as expected when considering results from earlier sections. These results suggest that compression is applicable for absolute thickness measurement as well as when the total change in thickness is much larger than a few micrometers.

6.8 Compressed traces and applications

As discussed in section 2.3 there are different applications to the Ultramonit system. When it comes to the high-rate corrosion experiments there are certainly no problem with compression; the relative uncertainty remains below 1% for bitrates down to 0.9 bit/sample. However, there are no pipelines which corrode with a yearly rate of 120 mm. Typical corrosion-rates are 0.1-1.0 mm/year. When looking at the low-rate corrosion experiment, the corrosion-rate is about 1.37 mm/year. Compressing these traces result in an uncertainty of +-0.2 mm/year, a value which is too high according to the requirement specification of +-0.1 mm/year. Even though 500 minutes "only" corresponds to 8.3 hours, extending the experiment to 12 hours (which corresponds to one work-shift) will not improve the results sufficiently.

It is clear that we have to consider the *total change in thickness* for a given application before compressing. In general one can assume that the uncertainty is inverse proportional to the time interval for which the measurements are taken. Hence, extending the low-rate experiment to 1000 minutes would probably yield an uncertainty of +-0.1 mm/year for the compressed traces, a value which is just within the system requirements.

When considering absolute thickness calculations, one can see from section 6.7 that bitrates of 1.4 bit/sample introduces at most and on average the additional uncertainties of 3 μm and 1 μm respectively. A more formal approach is presented in Appendix H. There it is demonstrated that for this bitrate, we can state with 99% certainty that the additional uncertainty due to compression is *at most* 2.3 μm. This amounts to only a fraction of the accepted uncertainty of +-0.1 mm and can be considered neglectable.

When considering the main application of long term monitoring of subsea pipelines, it is appropriate to employ the results from the high-rate traces, due to the corresponding orders of magnitude related to the change in thickness: While the total change in thickness for the high-rate corrosion experiment is about 100 μm, one can assume that the change in thickness between to consecutive measurement in a real operation is between 50-500 μm. Since we know that very little information is lost when compressing the high-rate traces, even for low bitrates, one can conclude that compression is applicable for an eventual real operation.

From this discussion we can conclude the following:

- Compression should in general not be employed if one aims at estimating the yearly corrosion-rate within +-0.1 mm during a work shift.
- For absolute thickness measurements or applications where the total change in thickness is much larger than the uncertainty introduced by the compression scheme, i.e. much larger than a few of micrometers, then

compression is applicable. This includes the main application of long term monitoring of subsea pipelines.

6.9 Summary and additional discussion of results

The results can be summarized as follows:

- The greedy algorithm is superior to the BWP-algorithm when it comes to processing time. Using the 7/9-filter before the 10/10-filter also cuts down on the processing time. Compared to transferring the unprocessed traces, the transfer time is reduced with a factor 12 at the optimal operating point.
- The preprocessing and zero-allocation of the high-rate traces affect the results to a minimal degree. The results related to the low-rate traces, however, demonstrate that the 10/10-filter performs better then the 7/9-filter. Even though the zero-allocation has a much larger negative effect on the output-fidelity than the preprocessing, an improved anti-aliasing filter should be considered.
- Estimating the distortion for the BWP-algorithm does not involve a degrading of the output.
- The BWP-algorithm outperforms the greedy algorithm when it comes to bitrate for a given fidelity. Results related to the high-rate corrosion experiment shows that bitrates of 0.9 bit/sample and 1.2 bit/sample (corresponding to compression ratios of 17.8 and 13.3) are obtainable while maintaining an acceptable fidelity. These results are closer connected to the run-length encoding scheme than the bit-allocation algorithms themselves. This is discussed more detailed in section 6.5.2.
- The dead-zone parameter should be optimized for each target bitrate.
- The low-rate traces are in general too sensitive for the proposed compression scheme. Preprocessing, however, does not degrade the fidelity at all if the anti-aliasing filter has sufficient performance. The latter approach yields a compression ratio of about 3.
- Compression increases the average uncertainty of the absolute thickness-estimate with 1-2 μm for bitrates between 1-1.4 bit/sample. This yields acceptable fidelity at the optimal operating point with respect to the main application of long term monitoring of subsea pipelines.
- A visual inspection of the compressed traces does in general not show any significant distortion. This applies for bitrates down to 0.9 bit/sample when incorporating the BWP-algorithm, and 1.2 bit/sample when using the greedy algorithm.

Even though the BWP-algorithm has proven to be too time-consuming compared to the greedy algorithm, it has given us important insights about the relation between the run-length encoding, the dead-zone parameter and the choice of nodes in the WP-tree. In general the fidelity of the output is very resistant to a large dead-zone parameter. This clears the way for long zero runs which in turn gives a better compression ratio. Furthermore, choosing nodes with better time-resolution yields a larger potential for long zero runs. Hence the BWP-algorithm has given us guidelines for improving our compression scheme.

From a merely practical viewpoint, the fact that one can decrease the transfer time with a factor 12 for the main application is appealing. As this involves employing the optimal operating point, we have in fact reached optimality regarding this application for the given encoding scheme. If we want to perform better, one must create an encoder with an optimal operating point corresponding to a lower bitrate, and simultaneously maintain the fidelity. A brief discussion of a suggested improvement follows in section 6.9.1.

6.9.1 Suggestion for improving the compression ratio for a given fidelity

Following the discussion in 6.5.2 related to run-length encoding, a small modification to the BWP-algorithm is performed so that only the nodes from the dyadic wavelet representation are regarded. Hence one gains processing time. Regarding the filtering process, the computational costs are about 47% compared to creating a full 4 level WP-tree. One can assume that the same results apply for the greedy algorithm which then yield processing times of about 22 ms. It is then possible to obtain optimality with respect to processing- and transfer time on operating points corresponding to bitrates of 0.7-0.8 bit/sample.

Some test cases have demonstrated that the modified approach does not decrease the fidelity significantly. For the high-rate experiments one easily reach a target bit-rate at 0.9 bit/sample while maintaining a good fidelity, while for the low-rate experiments, bitrates of about 1.3 bit/sample should give the same result as shown in Figure 40 a).

Note that this modification is not implemented onto the microcontroller, so the reduction to the processing time is only estimated. A more thorough study of this theme should be considered as future work.

7 Conclusions and future work

A relatively low-complexity encoding scheme, including two alternative bit allocation algorithms and filters, is successfully implemented onto the MicroBlaze microcontroller. It is argued that there is an intricate relation between transfer time, processing time, compression ratio and fidelity, where the relationship between transfer- and processing time is interpreted in terms of operating points. Given our encoding scheme, we have shown that output with acceptable fidelity is reached at the optimal operating point. This applies for the main application of long term monitoring of subsea pipelines. However, for ultra-high precision applications where the total change in thickness due to corrosion is less than a few micrometers, compression should not be employed. Preprocessing may be an alternative in the latter case though, reducing the transfer time with a factor of 2-3.

At the optimal operating point, the transfer time is reduced with a factor 12 compared to transferring uncompressed traces. This involves using the greedy algorithm together with the 7/9-filter. Even though one obtains better compression ratios for a given fidelity when using the BWP-algorithm, this is not as appealing due to the high processing time.

Concerning the proposed encoding scheme, the computational costs represent the constraint with respect to increased performance. Hence, future work should at first aim at decreasing the processing time while maintaining the fidelity. Some directions for future work follows:

- As the decoder has “unlimited” resources, one should aim at an asymmetric system for which as much complexity as possible is put in the decoder. There exist e.g. particular filter-banks which are suited for this.
- To improve the performance, especially with respect to the compressed low-rate traces, an improved anti-aliasing filter as well as the zero-allocation scheme suggested in section 6.4.2.2 should be implemented and tested on the microcontroller.
- The suggested improvement of section 6.9.1 should be implemented and tested on the microcontroller.
- The FPGA gives opportunities for HW-implementation of various components. This should be considered for the filter-bank in particular, as this is the most time-consuming building block.
- Entropy coding as well as better adaptivity (e.g. splitting the encoded data into blocks) should be considered in a future encoding scheme.
- A lossless compression scheme together with preprocessing should be considered for ultra-high precision applications, such as for the low-rate traces.
- Look into other methods for exploiting the highly non-linear approximation properties of wavelet decomposition schemes. It would e.g. be very interesting to test out variants of zero-tree encoding fit for one-dimensional signals.
- The results are based on a very limited choice of traces. In general there will be a larger uncertainty related to the form and quality of the received traces. The compression scheme should therefore be tested with more varied data.

8 References

[1] "Xilinx Spartan 3 overview," [Online] Available: http://www.xilinx.com/products/silicon_solutions/fpgas/spartan_series/spartan3_fpgas/overview.htm

[2] F. Vahid, "The softening of hardware," IEEE Computer society, April 2003, pp. 27-34

[3] "MicroBlaze product brief (PDF)," [Online] Available: http://www.xilinx.com/bvdocs/ipcenter/data_sheet/MB_sell_sheet.pdf

[4] J.D. Gibson, T. Berger, T. Lookabaugh, D. Lindbergh, R.L. Baker, "Digital Compression for Multimedia. Principles & Standards," Morgan Kaufman publishers, 1998

[5] Y. Linde, A. Buzo, R.M. Gray, "An algorithm for vector quantizer design," IEEE Trans. Comm., pp. 84-95, January 1980¨

[6] J.G. Proakis, D.G. Manolakis, "Digital Signal Processing. Principles, Algorithms, and Applications," 3rd edition, Prentice Hall, 1996

[7] V.K. Goyal, "Theoretical Considerations of Transform Coding," IEEE Signal Processing Magazine, September 2001

[8] K. Sayood, "Lossless Compression Handbook," Academic Press, 2003

[9] C. E. Shannon, "A mathematical theory of communication," Bell System Technical Journal, vol. 27, pp. 379-423 and 623-656, July and October, 1948.

[10] T.M. Cover, J.A. Thomas, "Elements of Information Theory," Wiley, New York, 1991

[11] S. Haykin, "Communication systems," 4th edition, John Wiley & Sons, Inc., 2001

[12] S. P. Lloyd, "Least squares quantization in PCM," IEEE Trans. Information Theory, pp. 127-135, March 1982

[13] A. Gersho, R.M. Gray, "Vector Quantization and Signal Compression," Kluwer Academic Publishers, 1992

[14] G. Øien, L. Lundheim, "Kompendium, TTT4125 Informasjonsteori, koding og kompresjon," 2004

[15] S. Mallat, "A wavelet tour of signal processing," 2nd edition, Academic Press, 1999

[16] H. Gish, J. Pierce, "Asymptotically efficient quantizing," IEEE Transactions on Information Theory, pp. 676-683, September 1968

[17] C. Christopoulos, A. Skodras, T. Ebrahimi, "The JPEG2000 Still Image Coding System: An Overview," IEEE Transactions on Consumer Electronics, Vol. 46, No. 45, pp. 1103-1127, November 2000

[18] S. Mallat, "Multiresolution approximations and wavelet orthonormal bases of $\mathbf{L}^2(\mathbb{R})$," Transactions of the American Mathematics Society, pp. 69-87, September 1989

[19] M. Smith, T. Barnwell, "Exact reconstruction for tree-structured subband coders," IEEE Transactions on Acoustics, Speech and Signal Processing, pp. 431-441, June 1986

[20] M. Vetterli, "Filter banks allowing perfect reconstruction," Signal Processing, Vol. 10, Nr. 3, pp. 219-244, 1986

[21] N. Young, "An introduction to Hilbert space," Cambridge University Press, 1988

[22] I. Daubechies, "Orthonormal bases of compactly supported wavelets," Communications on Pure and Applied Mathematics, pp. 909-996, November 1988
[23] G. Strang, T. Nguyen, "Wavelets and Filter Banks," Wellesley Cambridge Press, 1996
[24] B. Usevitch, "A tutorial on modern lossy image compression: Foundations of JPEG2000," IEEE Signal Processing Magazine, September 2001
[25] R. Coifman, Y. Mayer, M. Wickerhauser, "Wavelet analysis and signal processing," Wavelets and their Applications, pp. 153-178, 1992
[26] "ROOP Telsonic Ultrasonix homepage," [Online]. Available: http://www.rooptelsonic.com/products_ndt.html
[27] J. Barry, E. Lee and D. Messerschmitt, "Digital Communication," 3rd edition, Kluwer Academic Publishers, 2004
[28] M. Vetterli, "Wavelets, Approximation and Compression," IEEE Signal Processing Magazine, pp. 59-73, September 2001
[29] K. Ramchandran, M. Vetterli, "Best Wavelet Packet Bases in a Rate-Distortion Sense," IEEE Transactions on Image Processing, vol. 2, no. 2, pp 160-175, April 1993
[30] J. Nocedal and S.J. Wright, "Numerical optimization," Springer verlag, 1999
[31] "Wikipedia, the free encyclopedia," [Online]. Available: http://en.wikipedia.org/wiki/Convex_hull
[32] T.A. Ramstad, "Sub-band coder with a simple bit allocation algorithm: a possible candidate for digital mobile telephony," International Conference on Acoustics, Speech, and Signal Processing, pp. 203-207, IEEE Press, May 1982
[33] A. Segall, "Bit allocation and encoding for vector sources," IEEE Transactions on Information Theory, pp. 162-169, Volume 22, March 1976
[34] G. Wallace, "The JPEG still picture compression standard," ACM Computer Architecture News, 1992
[35] MATLAB documentation.
[36] E. Kreyszig, "Advanced Engineering Mathematics," 8th edition, John Wiley & Sons, Inc., pp. 1021, 1999
[37] J.Y. Huang, P.M. Schultheiss, "Block quantization of correlated Gaussian random variables," IEEE Transactions on Comm., September 1963
[38] K. Ramchandran, M. Vetterli and C. Herley, " Wavelets, Subband Coding and Best Bases," Proceedings of the IEEE, Vol. 84, No. 4, April 1996
[39] "MinGW homepage," [Online]. Available: http://www.mingw.org/

Appendix A. Encoder/Decoder

The encoding scheme is both implemented on the MicroBlaze microcontroller and on a standard PC. All the results concerning compressing 500 traces are performed on a PC for convenience. A standard C-compiler, *MinGW* [39], are employed. There are four source files implemented on the PC:

- *intLoop.c*. This version reads a file containing 500 traces and compresses them.
- *decLoop.c*. This version decompresses the 500 traces which is compressed with *intLoop.c*.
- *intVersion.c.* Identical to *intLoop.c* except for compressing one trace only.
- *decoder.c*. Identical to *decLoop.c* except for decompressing one trace only.

The parameters deciding the properties of the encoder are as follows:

- `USE_ESTIMATED_DISTORTION`: Define this if one employs the BWP-algorithm together with estimated distortion. Else the distortion is calculated. If the greedy-algorithm is used, this parameter is not taken into account.
- `NUM_VAR X/DEN_VAR Y`: Defines the bit allocation parameter for the greedy algorithm. If one e.g. wants a value of 3.5, X should equal 7 and Y should equal 2 (7/2=3.5). Recall that only integer arithmetic is used.
- `RBUDGET X`: Defines the rate budget where X is the total number of bits allowed to be used.
- `HIGH_RATE`: Define this if one wants to compress the high-rate traces. Otherwise the low-rate traces are compressed.

These parameters must be the same in the encoder and the decoder:

- `DEADZONE_NUM X/DEADZONE_DEN Y`: Defines the dead-zone parameter. If one e.g. wants a dead-zone parameter of 1.4, X should equal 7 and Y should equal 5.
- `USE_BIORTHOGONAL_WAVELETS`: Define this if the 7/9-filter is to be employed. Otherwise, the 10/10-filter is used.

The current source file for the microcontroller, *encoder_microblaze.c* will only compile and execute using the Xilinx Embedded Development Kit/Xilinx Platform Studio together with appropriate HW including the Xilinx Spartan3 FPGA and 1 MB SRAM. A program for communication through the COM-port, such as "Hyperterminal" for "Windows," must also be employed for dealing with the input/ output.

The input-files employed in this thesis are as follows:

- *testIn.enc*: Input file for *encoder_microblaze.c* and *intVersion.c*. This file includes one trace in binary format corresponding to 16 bits signed integers.
- *hrate_orig.dat*: Input file for *intLoop.c*. This file includes 500 traces from the high-rate corrosion experiment. They are represented in binary format corresponding to 16 bits signed integers.
- *lrate_original.dat*: Input file for *intLoop.c*. This file includes 500 traces from the low-rate corrosion experiment. They are represented in binary format corresponding to 16 bits signed integers.

Output-files:

- *testOut.dat*: Output file from *intLoop.c* and *intVersion.c*. This file must be copied to the directory where *decLoop.c* and *decoder.c* operates respectively.
- *testOutDec.dat*: Output file from *decLoop.c* and *decoder.c*

In order to read and interpolate the output of the decoder in MATLAB, the following commands are used:

Output of *decLoop.c*:

```
fid =fopen('testOutDec.dat','rb');
x=fread(fid,[1024 500],'double');
fclose(fid)
for i=1:size(x,2)
    xut(:,i)=interp(x(:,i),2);
end
```

Output of *decoder.c*:

```
fid =fopen('testOutDec.dat','rb');
x=fread(fid,1024 'double');
fclose(fid)
xut=interp(x,2);
```

Note that the only difference is that one reads in 500 instead of one trace when using *intLoop.c/ decLoop.c* instead of using *intVersion.c/ decoder.c.*

Appendix B. Orthogonal wavelets

An important aspect of signal decomposition is to find a good basis for the input signal. Our goal is orthogonal wavelet bases. In order to reach that goal we start with a definition of *multiresolution approximation* (MRA) [18] before we proceed with relations to the CMFs and the discrete wavelet transform.

Multiresolution approximation

With MRA one can approximate a function *f* at a resolution 2^{-j}. This approximation is a kind of local average of *f* over intervals proportional to the scale 2^j. Note that the scale is the inverse of the resolution, i.e. a larger scale gives a lower resolution and vica verca. Since we want orthogonal wavelet bases, the approximation of f at the resolution 2^{-j} is an orthogonal projection onto a space $\mathbf{V}_j$. The projection is denoted $P_{\mathbf{V}_j} f = f_j$. Recall that an orthogonal projection onto the space $\mathbf{V}_j$ selects the function $f_j \in \mathbf{V}_j$ which minimizes the squared norm $\| f - f_j \|^2 = \int_{\mathbb{R}} | f(t) - f_j(t) |^2 \, dt$. Thus it is an *optimal approximation* in a mse sense with respect to $\mathbf{V}_j$. More formally, consider the space $\mathbf{L}^2(\mathbb{R})$ of square integrable functions, that is $f(x) \in \mathbf{L}^2(\mathbb{R}) \Leftrightarrow \int_{\mathbb{R}} | f(x) |^2 \, dx < \infty$. A sequence $\{\mathbf{V}_j\}_{j \in \mathbb{Z}}$ of closed subspaces of $\mathbf{L}^2(\mathbb{R})$ is defined as a MRA if six different properties are satisfied. Note that some texts use the term multiresolution analysis instead. The properties are as follows:

$$
\begin{aligned}
&1: \forall (j,k) \in \mathbb{Z}^2, f(t) \in \mathbf{V}_j \Leftrightarrow f(t - 2^j k) \in \mathbf{V}_j \\
&2: \forall j \in \mathbb{Z}, \mathbf{V}_{j+1} \subset \mathbf{V}_j \\
&3: \forall j \in \mathbb{Z}, f(t) \in \mathbf{V}_j \Leftrightarrow f(\frac{t}{2}) \in \mathbf{V}_{j+1} \\
&4: \lim_{j \to -\infty} \mathbf{V}_j = \mathbf{L}^2(\mathbb{R}) \\
&5: \lim_{j \to \infty} \mathbf{V}_j = \{0\}
\end{aligned}
\qquad (10.1)
$$

6: *There exists a function* $\varphi \in \mathbf{V}_0$ *such that* $\{\varphi(t-n)\}_{n \in \mathbb{Z}}$ *is an orthonormal basis in* $\mathbf{V}_0$.

The first property states that the space $\mathbf{V}_j$ is translation invariant to the scale 2^j. Property 2 (causality property) means that an approximation in $\mathbf{V}_j$ contains all information for computing an approximation at a coarser resolution. The implication of this is that if one wants to approximate *f* with respect to $\mathbf{V}_{j+1}$, and already has the approximation f_j, then $P_{\mathbf{V}_{j+1}} f = P_{\mathbf{V}_{j+1}} f_j$. This is important for the DWT where one successively decomposes each approximation $P_{\mathbf{V}_j} f$ into coarser approximations $P_{\mathbf{V}_{j+1}} f$. The third property illustrates that if one dilates an approximation in $\mathbf{V}_j$ by 2, this is still defined as an approximation at the coarser resolution 2^{-j-1}. Property 4 means that every function in $\mathbf{L}^2(\mathbb{R})$ can be approximated to an arbitrary precision and property 5 states that one loses all the information about the function as j goes to ∞.

The sixth property is a bit simplified herein; $\{\varphi(t-n)\}_{n\in\mathbb{Z}}$ actually only needs to be a Riesz basis, a term not treated herein. The most important fact to know about Riesz bases, is that they can be transformed into an orthonormal basis. Since we only are interested in orthonormal bases, property six suffice. And since the basis is orthonormal, the following property holds:

$$< \varphi(t-n), \varphi(t-k) >= \delta(k-n) \tag{10.2}.$$

Here <.,.> and δ(.) denotes the inner product and the delta function respectively. The Fourier transform is defined as

$$F(\omega) = \int_{\mathbb{R}} f(t) e^{-jwt} dt \tag{10.3}.$$

The Parseval relation gives the relationship between an inner product of functions in the time- and frequeny domain:

$$< f(t), g(t) >= \frac{1}{2\pi} < F(\omega), G(\omega) > \tag{10.4}.$$

Here *F(ω)* and *G(ω)* is the Fourier transform of *f(t)* and *g(t)* respectively. By employing equation (10.4) with equation (10.2), one obtains, after some calculation, the necessary Fourier-domain-criterion for $\{\varphi(t-n)\}_{n\in\mathbb{Z}}$ to form an orthonormal system:

$$\sum_{m=-\infty}^{\infty} |\Phi(\omega+2\pi m)|^2 = 1 \tag{10.5}.$$

If one denotes

$$\varphi_{j,n}(t) = \frac{1}{\sqrt{2^j}} \varphi(\frac{t-2^j n}{2^j}) \tag{10.6}$$

it can be shown that $\{\varphi_{j,n}(t)\}_{n\in\mathbb{Z}}$ is an orthonormal basis of $\mathbf{V}_j$, for all j. The function φ in property six is called *the scaling function*. Thus the basis of $\mathbf{V}_j$ consists of dyadic shifts of its own scaling function.

Relations between the scaling function and the quadrature mirror filter

Since $\mathbf{V}_1 \subset \mathbf{V}_0$ the scaling function of $\mathbf{V}_1$ can be decomposed into a weighted sum of the orthonormal basis functions of $\mathbf{V}_0$:

$$\frac{1}{\sqrt{2}} \varphi(\frac{t}{2}) = \sum_{n=-\infty}^{\infty} h_n \varphi(t-n) \tag{10.7}.$$

Taking the inner product of both sides of equation (10.7) with *φ(t-n)* yields:

$$h_n = < \frac{1}{\sqrt{2}} \varphi(\frac{t}{2}), \phi(t-n) > \quad (10.8).$$

Equation (10.8) is called the *scaling function in time-domain.* The sequence h_n is interpreted as a filter, and it will be demonstrated that, as long as the DWT is defined for this scaling function, it actually is the DWT's low-pass filter! Defining the Fourier transform as

$$F(\omega) = \int_{\mathbb{R}} f(t) e^{-jwt} dt \quad (10.9)$$

and doing some calculations, gives the frequency-domain version of the scaling function:

$$\Phi(w) = \frac{1}{\sqrt{2}} \Phi(\frac{\omega}{2}) \sum_{n=-\infty}^{\infty} h_n e^{\frac{-j\omega n}{2}} = \frac{1}{\sqrt{2}} \Phi(\frac{\omega}{2}) H(\frac{\omega}{2}) \quad (10.10).$$

We see that *H(w)* is the *discrete-time Fourier transform* defined as

$$H(\omega) = \sum_{n=-\infty}^{\infty} h_n e^{-j\omega n} \quad (10.11).$$

Expanding equation (10.10) gives

$$\Phi(w) = \frac{1}{\sqrt{2}} H(\frac{\omega}{2}) \Phi(\frac{\omega}{2}) = \frac{1}{\sqrt{2}} H(\frac{\omega}{2}) \frac{1}{\sqrt{2}} H(\frac{\omega}{2^2}) \Phi(\frac{\omega}{2^2}) = \ldots$$
$$\Rightarrow \Phi(w) = \frac{1}{\sqrt{2}} H(\frac{\omega}{2}) \frac{1}{\sqrt{2}} H(\frac{\omega}{2^2}) \ldots \frac{1}{\sqrt{2}} H(\frac{\omega}{2^p}) \Phi(\frac{\omega}{2^p}) .$$

Letting $p \rightarrow \infty$ and stating without proof that $\Phi(0) = 1$ gives *the scaling relation* which defines MRA:

$$\Phi(w) = \prod_{j=1}^{\infty} \frac{1}{\sqrt{2}} H(\frac{\omega}{2^j}) \quad (10.12).$$

Plugging $\Phi(0) = 1$ into equation (10.10) leads to another property of *H(w)*, namely that $H(0) = \sqrt{2}$. To see the relation to CMFs, plug the right hand side of equation (10.10) into equation (10.5) which gives

$$1 = \sum_{n=-\infty}^{\infty} |\Phi(\omega + 2\pi n)|^2 = \frac{1}{2} \sum_{n=-\infty}^{\infty} |\Phi(\frac{\omega}{2} + n\pi)|^2 |H(\frac{\omega}{2} + n\pi)|^2$$
$$= \frac{1}{2} \sum_{l=-\infty}^{\infty} |\Phi(\frac{\omega}{2} + 2l\pi)|^2 |H(\frac{\omega}{2} + 2l\pi)|^2 + \quad (10.13).$$
$$\frac{1}{2} \sum_{l=-\infty}^{\infty} |\Phi(\frac{\omega}{2} + (2l+1)\pi)|^2 |H(\frac{\omega}{2} + (2l+1)\pi)|^2$$

The only operation performed above is that the original sum is split in two sums over odd and even numbers. Since $H(\omega)$ is 2π – periodic,

$$H(\frac{\omega}{2}+2l\pi)=\sum_{n=-\infty}^{\infty}h_n e^{-jn(\frac{\omega}{2}+2l\pi)}=H(\frac{\omega}{2}),$$
$$H(\frac{\omega}{2}+(2l+1)\pi)=\sum_{n=-\infty}^{\infty}h_n e^{-jn(\frac{\omega}{2}+(2l+1)\pi)}=H(\frac{\omega}{2}+\pi) \quad (10.14)$$

and we see that $H(.)$ is independent of l. Thus equation (10.13) can be rewritten as

$$1=\frac{1}{2}|H(\frac{\omega}{2})|^2\sum_{l=-\infty}^{\infty}|\Phi(\alpha+2l\pi)|^2+\frac{1}{2}|H(\frac{\omega}{2}+\pi)|^2\sum_{l=-\infty}^{\infty}|\Phi(\beta+2l\pi)|^2 \quad (10.15).$$

where the change of variables $\alpha=\frac{\omega}{2},\beta=\frac{\omega}{2}+\pi$ is employed. Recalling once more that $\sum_{n=-\infty}^{\infty}|\Phi(\omega+2\pi n)|^2=1$ and plugging this into equation (10.15) gives the desired CMF property (see e.g. [6]):

$$|H(\omega)|^2+|H(\omega+\pi)|^2=2 \quad (10.16).$$

This property is, as mentioned before, important for perfect reconstruction.

The wavelet function

Since $\ldots\subset\mathbf{V}_{-j}\subset\mathbf{V}_{-j+1}\subset\ldots\mathbf{V}_0\subset\mathbf{V}_{-1}\ldots\subset\mathbf{V}_j\subset\ldots$, and especially since $\mathbf{V}_0\subset\mathbf{V}_{-1}$, there must exist a space $\mathbf{W}_0$ orthogonal to $\mathbf{V}_0$ such that $\mathbf{V}_0\oplus\mathbf{W}_0=\mathbf{V}_{-1}$ where $\oplus$ is the orthogonal direct sum operator [21]. The implication of this is that $\varphi(t)\in\mathbf{V}_0,\psi(t)\in\mathbf{W}_0\Rightarrow<\varphi(t),\psi(t)>=0$. $\psi(t)$ is denoted the wavelet function. By noting that

$$\mathbf{V}_L=\mathbf{V}_{L+1}\oplus\mathbf{W}_{L+1}=\mathbf{V}_{L+2}\oplus\mathbf{W}_{L+2}\oplus\mathbf{W}_{L+1}=\oplus_{j=L+1}^{j=J}\mathbf{W}_j\oplus\mathbf{V}_J \quad (10.17)$$

and using properties number 4 and 5 from the definition of the MRA we see that, when L and J go to -∞ and ∞ respectively, that $\mathbf{W}_j$ is orthogonal to $\mathbf{W}_n$ for any $j,n\in\mathbb{Z},j\neq n$ and that:

$$\oplus_{j=-\infty}^{j=\infty}\mathbf{W}_j=\mathbf{L}^2(\mathbb{R}) \quad (10.18).$$

Similarly as for the scaling function it is possible to prove that for any scale 2^j, the set of functions $\{\psi_{j,n}(t)\}_{n\in\mathbb{Z}}$ is an orthonormal basis of $\mathbf{W}_j$. Here $\psi_{j,n}(t)$ is denoted as:

$$\psi_{j,n}(t)=\frac{1}{\sqrt{2^j}}\psi(\frac{t-2^j n}{2^j}) \quad (10.19).$$

But for the wavelet function one can also see from equation (10.18) that $\{\psi_{j,n}(t)\}_{j\in\mathbb{Z},n\in\mathbb{Z}}$ is an orthonormal basis of $\mathbf{L}^2(\mathbb{R})$. The orthonormality condition for the wavelet function in the Fourier-domain is similar as for the scaling function:

$$\sum_{m=-\infty}^{\infty} |\Psi(\omega+2\pi m)|^2 = 1 \tag{10.20}$$

Since $\mathbf{W}_0$ clearly is a subspace of $\mathbf{V}_{-1}$ one can, by following the same procedure as in for the scaling function, express $\psi(t)$ as a series of orthonormal basis functions:

$$\psi(t) = \sum_{n=-\infty}^{\infty} g_n \sqrt{2}\varphi(2t-n) \tag{10.21}.$$

The sequence $\{g_n\}$ is just like $\{h_n\}$ interpreted as a filter, and this time as the DWT's high-pass filter. In the frequency domain equation (10.21) becomes

$$\Psi(w) = \frac{1}{\sqrt{2}}\Phi(\frac{\omega}{2})\sum_{n=-\infty}^{\infty} g_n e^{\frac{-j\omega n}{2}} = \frac{1}{\sqrt{2}}\Phi(\frac{\omega}{2})G(\frac{\omega}{2}) \tag{10.22}$$

By using the same trick as we did for getting the CMF property of $H(\omega)$ (this time by plugging equation (10.22) into equation (10.20), see calculations in (10.13)-(10.16)), one obtains the same result for the filter $G(\omega)$:

$$|G(\omega)|^2 + |G(\omega+\pi)|^2 = 2 \tag{10.23}.$$

Orthogonality between Wj and Vj and relations to H(ω) and G(ω)

In time-domain, orthogonality between $\mathbf{W}_0$ and $\mathbf{V}_0$ means that

$$<\psi(t), \varphi(t-n)>= 0, n\in\mathbb{Z} \tag{10.24}.$$

Using the Parseval relation on equation (10.24) yields:

$$\begin{aligned} <\psi(t),\varphi(t-n)> &= \int_{-\infty}^{\infty}\psi(t)\varphi(t-n)^* dt = \frac{1}{2\pi}\int_{-\infty}^{\infty}\Psi(\omega)\Phi(\omega)^* e^{jn\omega}d\omega \\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi}[\sum_{k=-\infty}^{\infty}\Psi(\omega+2k\pi)\Phi(\omega+2k\pi)^*]e^{jn\omega}d\omega = 0 \end{aligned} \tag{10.25}.$$

Thus the integrand has to be zero:

$$\sum_{k=-\infty}^{\infty}\Psi(\omega+2k\pi)\Phi^*(\omega+2k\pi) = 0 \tag{10.26}.$$

The * - superscript in the above equation denotes complex conjugate. By inserting equation (10.10) and (10.22) into equation (10.26) and splitting the even and odd terms once more, one obtains:

$$
\begin{aligned}
0 &= \sum_{k=-\infty}^{\infty} \Psi(\omega+2k\pi)\Phi^*(\omega+2k\pi) \\
&= \sum_{k=-\infty}^{\infty} \frac{1}{2} G(\frac{\omega+2k\pi}{2})\Phi(\frac{\omega+2k\pi}{2})H^*(\frac{\omega+2k\pi}{2})\Phi^*(\frac{\omega+2k\pi}{2}) \\
&= \frac{1}{2}\sum_{k=-\infty}^{\infty} G(\frac{\omega+2k\pi}{2})H^*(\frac{\omega+2k\pi}{2})\,|\,\Phi(\frac{\omega+2k\pi}{2})\,|^2 \\
&= \frac{1}{2}\sum_{m=-\infty}^{\infty} G(\frac{\omega}{2}+2m\pi)H^*(\frac{\omega}{2}+2m\pi)\,|\,\Phi(\frac{\omega}{2}+2m\pi)\,|^2 \\
&+ \frac{1}{2}\sum_{m=-\infty}^{\infty} G(\frac{\omega}{2}+2m\pi+\pi)H^*(\frac{\omega}{2}+2m\pi+\pi)\,|\,\Phi(\frac{\omega}{2}+2m\pi+\pi)\,|^2 \\
&= \frac{1}{2}G(\frac{\omega}{2})H^*(\frac{\omega}{2}) + \frac{1}{2}G(\frac{\omega}{2}+\pi)H^*(\frac{\omega}{2}+\pi)
\end{aligned}
\qquad (10.27).
$$

In the last equality, the fact that $H(\omega)$ and $G(\omega)$ are 2π-periodic functions and the fact that $\sum_{n=-\infty}^{\infty} |\Phi(\omega+2\pi n)|^2 = 1$ is used. Even though we started with the spaces $\mathbf{W}_0$ and $\mathbf{V}_0$, any two spaces $\mathbf{W}_j$ and $\mathbf{V}_j$ would give the same result. Since ω is arbitrary, we obtain the desired form which relates orthogonality between $\mathbf{W}_j$ and $\mathbf{V}_j$ to the filters $H(\omega)$ and $G(\omega)$:

$$G(\omega)H^*(\omega) + G(\omega+\pi)H^*(\omega+\pi) = 0 \qquad (10.28).$$

By choosing

$$G(\omega) = e^{-j\omega}H^*(\omega+\pi) \qquad (10.29)$$

both (10.23) and (10.28) is fulfilled. Thus we see that the filter $G(\omega)$ is entirely characterized by $H(\omega)$. Taking the inverse Fourier transform of equation (10.29) yields:

$$g_n = (-1)^{1-n} h_{1-n} \qquad (10.30).$$

and we have the same relation in time-domain.

The discrete wavelet transform

MRA is about approximating a function on different resolutions 2^{-j}. By employing the wavelet-function one can also keep the detailed information which is lost during the approximation from $\mathbf{V}_j$ to $\mathbf{V}_{j+1}$. The approximations- and wavelet coefficients on the resolution 2^{-j+1} are obtained by orthogonal projections down to the spaces $\mathbf{V}_{j+1}$ and $\mathbf{W}_{j+1}$. From now on the wavelet coefficients are denoted detail coefficients. By denoting the approximation- and detail coefficients at resolution 2^{-j+1} for $a_{j+1}(n)$ and $d_{j+1}(n)$ respectively, we have:

$$a_{j+1}(p) =< f, \varphi_{j+1,p} > \\ d_{j+1}(p) =< f, \psi_{j+1,p} > \tag{10.31}.$$

Since $\mathbf{V}_{j+1} \oplus \mathbf{W}_{j+1} = \mathbf{V}_j$, $\varphi_{j+1,p}(t) \in \mathbf{V}_{j+1}$ and $\psi_{j+1,p}(t) \in \mathbf{W}_{j+1}$ can be decomposed in the orthonormal basis functions $\{\varphi_{j,n}\}_{n\in\mathbb{Z}}$ which span $\mathbf{V}_j$:

$$\varphi_{j+1,p}(t) = \sum_{n=-\infty}^{\infty} < \varphi_{j+1,p}, \varphi_{j,n} > \varphi_{j,n}(t) \\ \psi_{j+1,p}(t) = \sum_{n=-\infty}^{\infty} < \psi_{j+1,p}, \varphi_{j,n} > \varphi_{j,n}(t) \tag{10.32}.$$

The inner products in (10.32) can be rewritten as follows with the change of variables $t = (\tau + 2p)2^j \Rightarrow dt = 2^j d\tau$:

$$< \varphi_{j+1,p}, \varphi_{j,n} >= \int_{-\infty}^{\infty} \frac{1}{\sqrt{2^{j+1}}} \varphi(\frac{t - 2^{j+1}p}{2^{j+1}}) \frac{1}{\sqrt{2^j}} \varphi^*(\frac{t - 2^j n}{2^j}) dt \\ = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2}} \varphi(\frac{\tau}{2}) \varphi^*(\tau - n + 2p) d\tau =< \frac{1}{\sqrt{2}} \varphi(\frac{\tau}{2}), \varphi(\tau - n + 2p) > \tag{10.33}.$$

Comparing (10.33) with equation (10.8) shows that the above inner products actually represent the filter coefficients:

$$< \varphi_{j+1,p}, \varphi_{j,n} >=< \frac{1}{\sqrt{2}} \varphi(\frac{\tau}{2}), \varphi(\tau - n + 2p) >= h_{n-2p} \tag{10.34}.$$

Using the same method for $\psi_{j+1,p}(t)$ gives a similar result:

$$< \psi_{j+1,p}, \varphi_{j,n} >= g_{n-2p} \tag{10.35}.$$

Now, taking the inner product of the function *f* with both sides of (10.32), using (10.31) and plugging in the filter coefficients for the inner products gives:

$$a_{j+1}(p) =< f, \sum_{n=-\infty}^{\infty} h_{n-2p} \varphi_{j,n}(t) >= \sum_{n=-\infty}^{\infty} h_{n-2p} a_j(n) \\ d_{j+1}(p) =< f, \sum_{n=-\infty}^{\infty} g_{n-2p} \varphi_{j,n}(t) >= \sum_{n=-\infty}^{\infty} g_{n-2p} a_j(n) \tag{10.36}.$$

Equation (3.17) defines the DWT (also called the fast wavelet transform) which obviously is a convolution. What is remarkable about this result is that the projection onto the spaces $\mathbf{V}_j$ and $\mathbf{W}_j$ turns out to be a filtering process which can be implemented very effectively on computers in either software or hardware. Looking more closely at (3.17) we see that the equation defines downsampling with a factor 2 as well as filtering, so it actually defines a decimation process.

Since $\mathbf{V}_{j+1} \oplus \mathbf{W}_{j+1} = \mathbf{V}_j$, any $\varphi_{j,p}$ can be written as

$$\varphi_{j,p} = \sum_{n=-\infty}^{\infty} < \varphi_{j,p}, \varphi_{j+1,n} > \varphi_{j+1,n}(t) + \sum_{n=-\infty}^{\infty} < \varphi_{j,p}, \psi_{j+1,n} > \psi_{j+1,n}(t) \qquad (10.37).$$

Inserting (10.34) and (10.35) and taking the inner product with *f* on both sides of (10.37) yields:

$$a_j(p) = \sum_{n=-\infty}^{\infty} h_{p-2n} a_{j+1}(n) + \sum_{n=-\infty}^{\infty} g_{p-2n} d_{j+1}(n) \qquad (10.38).$$

Equation (10.38) defines the inverse DWT (IDWT) and demonstrates that one obtains perfect reconstruction. The above equation shows that the IDWT is an interpolation process, i.e. it upsamples with 2 before filtering.

Daubechies scaling functions and wavelets

Some orthogonal wavelets/scaling functions are presented in the figure below:

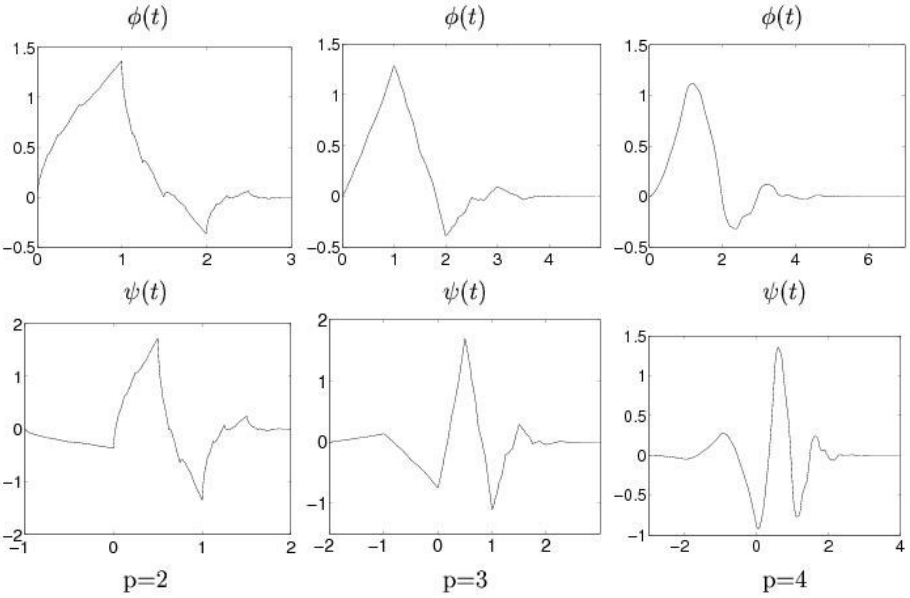


Figure 42 The Daubechies scaling function φ and wavelet ψ for p = 2, p = 3 and p = 4.

Example - Construction of Haar system from MRA

In this example, we will construct the filters *H(ω)* and *G(ω)* from the simple Haar scaling function defined as:

$$\begin{aligned} \varphi(t) &= 1, 0 < t < 1 \\ \varphi(t) &= 0, otherwise \end{aligned} \qquad (10.39).$$

It can be shown that φ satisfies all the MRA properties (see (10.1)). Knowing that $\phi(t) \in \mathbf{V}_0$, one also knows from equation (10.6) that $\frac{1}{\sqrt{2}}\varphi(\frac{t}{2}) \in \mathbf{V}_1$. Since

$$\begin{aligned} &\frac{1}{\sqrt{2}}\varphi(\frac{t}{2}) = \frac{1}{\sqrt{2}}, 0 < t < 2 \\ &\frac{1}{\sqrt{2}}\varphi(\frac{t}{2}) = 0, otherwise \end{aligned} \tag{10.40}$$

and using (3.9) we see that equation (10.40) can be expanded in the series

$$\frac{1}{\sqrt{2}}\varphi(\frac{t}{2}) = \frac{1}{\sqrt{2}}(\varphi(t) + \varphi(t-1)) \tag{10.41}.$$

Thus we know that

$$\begin{aligned} &h_0 = h_1 = \frac{1}{\sqrt{2}} \\ &h_l = 0 \forall l \neq \{0,1\} \end{aligned} \tag{10.42}.$$

Taking the discrete-time Fourier transform defined in (10.11) yields

$$H(\omega) = \frac{1}{\sqrt{2}}(1 + e^{-j\omega}) = \sqrt{2}e^{-j\frac{\omega}{2}}\cos(\frac{\omega}{2}) \tag{10.43}.$$

Using equation (3.15) we see that

$$\begin{aligned} &g_0 = -g_1 = -\frac{1}{\sqrt{2}} \\ &g_l = 0 \forall l \neq \{0,1\} \end{aligned} \tag{10.44}$$

which in the Fourier domain becomes

$$G(\omega) = \sqrt{2}e^{-j(\frac{\pi+\omega}{2})}\sin(\frac{\omega}{2}) \tag{10.45}.$$

Clearly $G(\omega)$ is a high-pass filter, while $H(\omega)$ is a low-pass filter. Now one can use equation (3.11) to construct the Haar wavelet:

$$\psi(t) = \begin{Bmatrix} -1, 0 < t < \frac{1}{2} \\ 1, \frac{1}{2} < t < 1 \\ 0, otherwise \end{Bmatrix} \tag{10.46}.$$

The Haar system is the simplest of the orthogonal wavelet systems. In general, the creation of such filters is much more complex; see e.g. [38].

Appendix C. Biorthogonal wavelets

In the orthogonal case, one uses the same filter (only time-reversed) for both analysis and synthesis. It is then impossible to obtain symmetric or antisymmetric FIR CMFs. In the biorthogonal case, one defines two MRAs (instead of one in the orthogonal case) of $\mathbf{L}^2(\mathbb{R})$, defined by the sequence of spaces $\{\mathbf{V}_j\}_{j\in\mathbb{Z}}$ and $\{\tilde{\mathbf{V}}_j\}_{j\in\mathbb{Z}}$. Again the functions $\{\varphi_{j,n}\}_{n\in\mathbb{Z}}$ span $\mathbf{V}_j$ and $\{\tilde{\varphi}_{j,n}\}_{n\in\mathbb{Z}}$ span $\tilde{\mathbf{V}}_j$. The biorthogonal condition is defined as

$$< \phi_{j,0}, \tilde{\phi}_{j,n} >= \delta_n \tag{11.1}.$$

The two wavelet families $\{\psi_{j,n}\}_{j,n\in\mathbb{Z}}$ and $\{\tilde{\psi}_{j,n}\}_{j,n\in\mathbb{Z}}$ which come from the MRAs, satisfies the following biorthogonal condition:

$$< \psi_{a,b}, \tilde{\psi}_{c,d} >= \delta_{a-c}\delta_{b-d} \tag{11.2}.$$

In this case, we loosen up on the orthogonality constraint between $\mathbf{V}_j$ and $\mathbf{W}_j$ and $\tilde{\mathbf{V}}_j$ and $\tilde{\mathbf{W}}_j$. The relations $\mathbf{V}_{j+1} \oplus \mathbf{W}_{j+1} = \mathbf{V}_j$ and $\tilde{\mathbf{V}}_{j+1} \oplus \tilde{\mathbf{W}}_{j+1} = \tilde{\mathbf{V}}_j$ still hold but now the $\oplus$-operator only denotes the *direct sum* [21], and not the orthogonal direct sum as for the orthogonal wavelets. The following orthogonality conditions between spaces are required though:

$$\mathbf{V}_j \perp \tilde{\mathbf{W}}_j, \tilde{\mathbf{V}}_j \perp \mathbf{W}_j \tag{11.3}.$$

The filter-conditions for perfect reconstruction are in this case

$$\begin{aligned} &H^*(\omega)\tilde{H}(\omega) + H^*(\omega+\pi)\tilde{H}(\omega+\pi) = 2 \\ &G(\omega) = e^{-jw}\tilde{H}^*(\omega+\pi), \tilde{G}(\omega) = e^{-jw}H^*(\omega+\pi) \end{aligned} \tag{11.4}.$$

Biorthogonal wavelets and scaling functions

The wavelets and scaling functions corresponding to the 7- and 9-taps JPEG2000 filters are presented in the figure below.

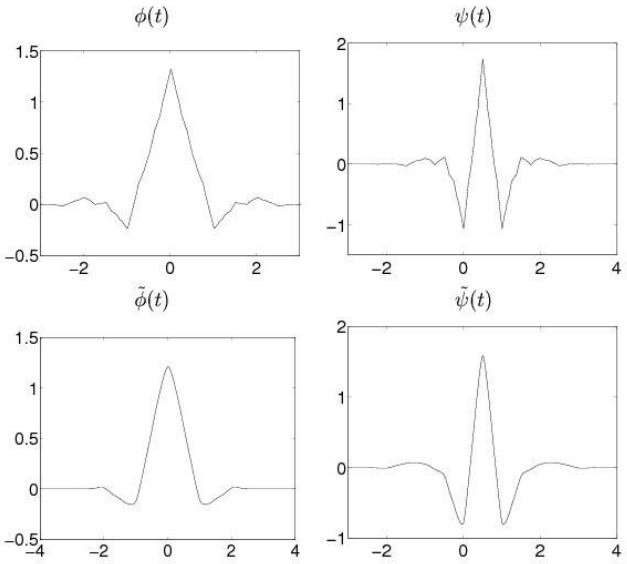


Figure 43 Biorthogonal wavelets and scaling functions. These are calculated with a 7-tap and a 9-tap filter which fulfill the biorthogonal conditions. The upper functions are used in the analysis phase, and the lower are used in the synthesis.

Appendix D. Signal decomposition filters

This appendix presents the frequency responses of the two signal decomposition filters in Figure 44 as well as their coefficients in Table 2.

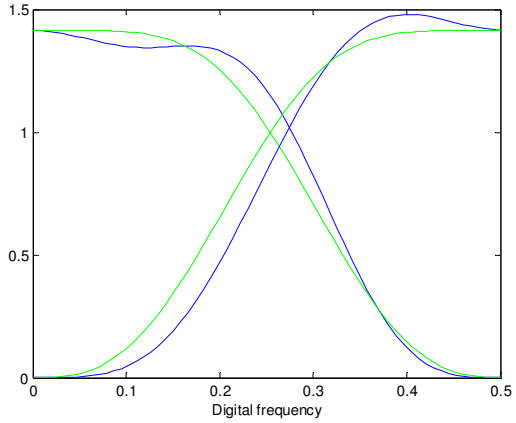


Figure 44 LP- and HP-characteristics of the 7/9-filter (blue) and 10/10-filter (green).

Table 2 LP filter coefficients for 10/10- and 7/9-filter.

Biorthogonal 9-tap	Biorthogonal 7-tap	Orthogonal 10-tap
0.037828455507264	-0.0645388826286971	0.160102397974125
-0.0238494650195568	-0.0406894176091641	0.603829269797473
-0.110624404418437	0.418092273221617	0.724308528438574
0.377402855612831	0.788485616405583	0.138428145901103
0.852698679008894	0.418092273221617	-0.24229488706619
0.377402855612831	-0.0406894176091641	-0.0322448695850295
-0.110624404418437	-0.0645388826286971	0.0775714938400651
-0.0238494650195568		-0.00624149021301171
0.037828455507264		-0.0125807519990155
		0.00333572528500155

Note the symmetric biorthogonal filter coefficients which makes it possible to deal with both boundary artifacts as well as coefficient expansion discussed in section 3.3.2. Also recall that the HP-filter coefficients easily are obtained from equation (3.15).

Appendix E. Signal decomposition and distortion

Our goal is to prove that the signal has to be projected down to orthogonal spaces in order to preserve the mse-distortion across scales.

Now, let $y = Q(x)$ where $Q(.)$ is the quantization operation. Also, let y and x be members of the real inner product space $\mathbf{V}_0$. Assume further that $\mathbf{V}_1 \oplus \mathbf{W}_1 = \mathbf{V}_0$ and that the functions $\{\varphi_n\}_{n\in\mathbb{Z}}$ are an orthonormal basis of $\mathbf{V}_1$, and the functions $\{\psi_n\}_{n\in\mathbb{Z}}$ are an orthonormal basis of $\mathbf{W}_1$. We assume nothing about orthogonality between $\mathbf{V}_1$ and $\mathbf{W}_1$ yet. Now y and x can be expanded as follows:

$$\begin{aligned} x &= \sum_k < x, \varphi_k > \varphi_k + < x, \psi_k > \psi_k \\ y &= \sum_k < y, \varphi_k > \varphi_k + < y, \psi_k > \psi_k \end{aligned} \tag{12.1}.$$

The distortion, $D_{\mathbf{V}_0}$, is calculated as follows:

$$\begin{aligned} D_{\mathbf{V}_0} &= \| x - y \|^2 = < x - y, x - y > = < x, x > -2 < x, y > + < y, y > \\ &= < \sum_k < x, \varphi_k > \varphi_k + < x, \psi_k > \psi_k, \sum_n < x, \varphi_n > \varphi_n + < x, \psi_n > \psi_n > \\ &-2 < \sum_k < x, \varphi_k > \varphi_k + < x, \psi_k > \psi_k, \sum_n < y, \varphi_n > \varphi_n + < y, \psi_n > \psi_n > \\ &+ < \sum_k < y, \varphi_k > \varphi_k + < y, \psi_k > \psi_k, \sum_n < y, \varphi_n > \varphi_n + < y, \psi_n > \psi_n > \end{aligned} \tag{12.2}.$$

The distortions, $D_{\mathbf{V}_1}$ and $D_{\mathbf{W}_1}$, can be calculated as

$$\begin{aligned} D_{\mathbf{V}_1} &= \| \sum_k < x, \varphi_k > \varphi_k - \sum_n < y, \varphi_n > \varphi_n \|^2 \\ &= \sum_k < x, \varphi_k >^2 -2 < x, \varphi_k >< y, \varphi_k > + < y, \varphi_k >^2 \\ D_{\mathbf{W}_1} &= \| \sum_k < x, \psi_k > \psi_k - \sum_n < y, \psi_n > \psi_n \|^2 \\ &= \sum_k < x, \psi_k >^2 -2 < x, \psi_k >< y, \psi_k > + < y, \psi_k >^2 \end{aligned} \tag{12.3}.$$

If the distortion is to be preserved across scales, the following relation must hold:

$$D_{\mathbf{V}_0} = D_{\mathbf{W}_1} + D_{\mathbf{V}_1} \tag{12.4}.$$

It can be easily verified, by calculating the inner products of (12.2), that the only possibility of equation (12.4) to hold, is if for $\forall m, n \in \mathbb{Z}, < \varphi_n, \psi_m > = 0$ which implies orthogonality between the spaces between $\mathbf{V}_1$ and $\mathbf{W}_1$.

Appendix F. Pseudo code for the fast convex search algorithm

This section presents a pseudo code for the optimization process described in section 5.4.1. The complete algorithm consists of an initialization phase, an outer loop which solves equation (5.6) and an inner loop which solves equation (5.4) each time λ is updated.

Initialization: Create the complete WP-tree with the algorithm described in section 3.3.3, and gather for each leaf the quantizer dependent operating points $(D_q(n_i), R_q(n_i))$, $\forall q \in \mathbf{q}(n_i)$. Moreover, associate the data structure $\{\tilde{R}, \tilde{D}, \tilde{J}, split\}$ with each $n_i \in T$.

Outer loop; finding optimal λ:

1. Pick $\lambda_l \le \lambda_u$ such that the rates corresponding to the solutions of (5.4), $R^*(\lambda_l), R^*(\lambda_u)$, fulfills $R^*(\lambda_u) \le R_{Budget} \le R^*(\lambda_l)$. If there is an equality on either side, we have the solution → stop optimizing. Otherwise proceed to step 2.
2. Pick $\lambda_{next} = (D(\lambda_u) - D(\lambda_l))/(R(\lambda_l) - R(\lambda_u)) + \varepsilon$ where ε is a small positive number which ensures that the lower rate operating point is chosen if $W(\lambda)$ has two maximum points. This guarantees that $\lambda_l \le \lambda_{next} \le \lambda_u$.
3. Find the best admissible tree and quantizers given λ_{next}. This is done with the inner loop procedure described below.
   Now, if $R^*(\lambda_{next}) = R^*(\lambda_u)$, then stop. $\lambda = \lambda^*$.
   Else if $R^*(\lambda_{next}) > R_{Budget}$, $\lambda_l \leftarrow \lambda_{next}$. Go to step 2.
   Else $\lambda_u \leftarrow \lambda_{next}$. Go to step 2.

Thus the outer loop makes the distance between λ_l and λ_u successively smaller until the optimal lambda is found.

Inner loop; find the best admissible tree and quantizers for a fixed λ:

1. Populate all nodes in the tree with the quantizers corresponding to their minimum costs, i.e $J_{n_i}(\lambda) = \min_q D_q(n_i) + \lambda R_q(n_i)$.
2. Initialize $k = d$ where d is the maximum tree-depth. For all nodes at maximum depth initialize the data structure; $\tilde{R} \leftarrow R_{q^*}(n_i), \tilde{D} \leftarrow D_{q^*}(n_i), \tilde{J} \leftarrow J_{n_i}(\lambda)$ and $split \leftarrow FALSE$.
3. $k \leftarrow k - 1$. If $k < 0$ go to step 6.
4. For all nodes n_i at depth k:
   If $J_{n_i}(\lambda) \le \tilde{J}_{left\_child} + \tilde{J}_{right\_child}$

then $\{split \leftarrow FALSE, \tilde{R} \leftarrow R_{q^*}(n_i), \tilde{D} \leftarrow D_{q^*}(n_i), \tilde{J} \leftarrow J_{n_i}(\lambda)\}$

else $\{split \leftarrow TRUE, \tilde{R} \leftarrow \tilde{R}_{left\_child} + \tilde{R}_{right\_child}, \tilde{D} \leftarrow \tilde{D}_{left\_child} + \tilde{D}_{right\_child}, \tilde{J} \leftarrow \tilde{J}_{left\_child} + \tilde{J}_{right\_child}\}$

5. Go to step 3.
6. Starting from the root node and traversing throughout the tree, carve out the optimal admissible tree $S^*(\lambda)$ and its associated choice of quantizers by using the parent-child relationship between the nodes. If one reaches a node where *split* equals FALSE, this is clearly a terminal node in the optimal admissible tree and its children are not further considered.

The inner loop optimization is based on *Bellman's optimality principle* [36] which states that if one knows the optimal admissible tree from node *n* onwards to the maximum tree-depth, then all surviving paths through node *n* must invoke this same optimal finishing path.

Appendix G. Optimal bit allocation for subbands using pdf - optimized quantizers

Obviously each of the K subbands has to be quantized in order to achieve the requested compression ratio. The problem of bit allocation is then to decide, given a rate-budget R_{Budget}, the number of bits for each subband, R_i, which minimizes the overall distortion, D, of the reconstructed signal, or mathematically speaking:

$$\min_{\mathbf{R}} D(\mathbf{R}) = \min_{\mathbf{R}} \sum_{i=0}^{K-1} D_i(R_i) \tag{13.1}$$

subject to

$$\sum_{i=0}^{K-1} R_i \le R_{Budget} \tag{13.2}$$

Here $\mathbf{R} = [R_0, R_1, \ldots, R_{K-1}]$ and D_i is the distortion in subband number i. The distortion functions D_i are rarely known in practice, but assuming the high resolution approximation and pdf-optimized quantizers [12] we have that [13]

$$D_i(R_i) \approx \alpha_i 2^{-2R_i} \sigma_i^2 \tag{13.3}$$

where α_i and σ_i^2 is a parameter dependent on the probability distribution and the variance in subband number i respectively. The classical solution to this problem is given in [37] and is expressed by:

$$R_i = \overline{R} + \frac{1}{2} lb \frac{\sigma_i^2 \alpha_i}{(\prod_{i=0}^{K-1} \sigma_i^2)^{1/k} (\prod_{i=0}^{K-1} \alpha_i)^{1/k}} bits/symbol \tag{13.4}.$$

Here $\overline{R} = R_{Budget} / K$. The problem with this solution is the possibility of negative bit allocation for a given subband and the occurrences of non-integer bit allocations. These problems are addressed by, amongst others, Ramstad in [32]. The most important conclusion to draw from equation (13.4) is that the number of bits allocated to a subband should increase with its variance.

Appendix H. Additional uncertainty due to compression

500 traces result in 500 thickness values. Denote the thickness values $Y = [y_1, y_2 ... y_{500}]^T$ and $\hat{Y} = [\hat{y}_1, \hat{y}_2 ... \hat{y}_{500}]^T$ corresponding to the uncompressed and compressed traces respectively. Also denote the deviation between them for $X = Y - \hat{Y}$. The mean and standard deviation of *X* is estimated as

$$\begin{aligned} \mu_X &= \frac{1}{500}\sum_{n=1}^{500} x_n \\ \sigma_X &= \sqrt{\frac{1}{500}\sum_{n=1}^{500}(x_n - \mu_X)^2} \end{aligned} \tag{14.1}$$

Here *X* is assumed to have a Gaussian distribution. Due to the large number of samples (500), one can employ the central limit theorem and state that the above values more or less equal the true mean and standard deviation. Thus we can state with a 99% confidence interval that the deviation in absolute thickness between the uncompressed vs. compressed traces is in the interval

$$[\mu_X - z_{0.005}\sigma_X, \mu_X + z_{0.005}\sigma_X] \tag{14.2}.$$

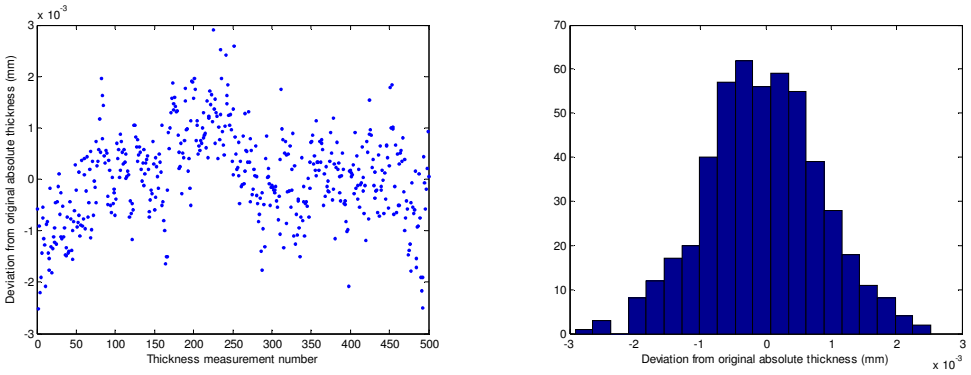
Here $z_{0.005}$ = 2.57 is the value corresponding to the area less then 0.005 for the normalized normal distribution. Now, let $\pm k$ mm be the uncertainty of the absolute thickness measurements from the unprocessed traces. Thus we can state with at least 99% confidence that the uncertainty of the compressed traces won't exceed $\pm(k + \max\{abs(\mu_X - z_{0.005}\sigma_X), abs(\mu_X + z_{0.005}\sigma_X)\})$ mm.

Let the uncertainty in absolute thickness due to compression only (not including the original uncertainty *k*) be defined as

$$u_C = \max\{abs(\mu_X - z_{0.005}\sigma_X), abs(\mu_X + z_{0.005}\sigma_X)\}) \tag{14.3}.$$

The following example illustrates the uncertainty related to the absolute thickness after compression. Parameters corresponding to a bitrate of 1.4 bit/sample are chosen:

- Filter type: 7/9
- Rate budget: 3300 bits
- Corrosion type: High-rate
- Dead-zone: 3
- Bit allocation parameter: 4
- Bit allocation algorithm: Greedy



a) b)

Figure 45 Deviation from original absolute thickness in a) and the corresponding historgram in b).

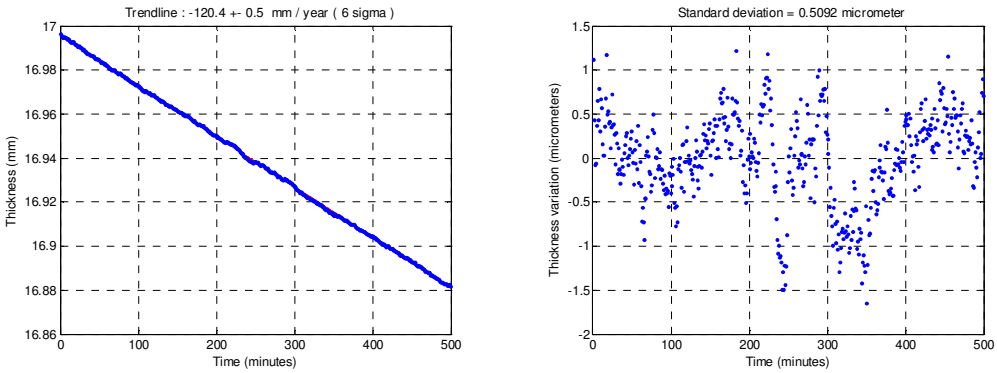
We can see from Figure 45 a) and b) that it is reasonable to assume a Gaussian distribution. Using these results, the mean and standard deviation becomes -0.047 µm and 0.88 µm respectively. The 99% confidence interval thus becomes $[-2.3\mu m, 2.2\mu m]$. This implies that the additional uncertainty to the absolute thickness is maximum u_C = 2.3 µm after compression.

Appendix I. Additional results

Trendlines and standard deviations after preprocessing and zero-allocation

Parameter set 1:

- Filter type: 7/9
- Corrosion type: High-rate
- Method for calculating thickness: Inversion



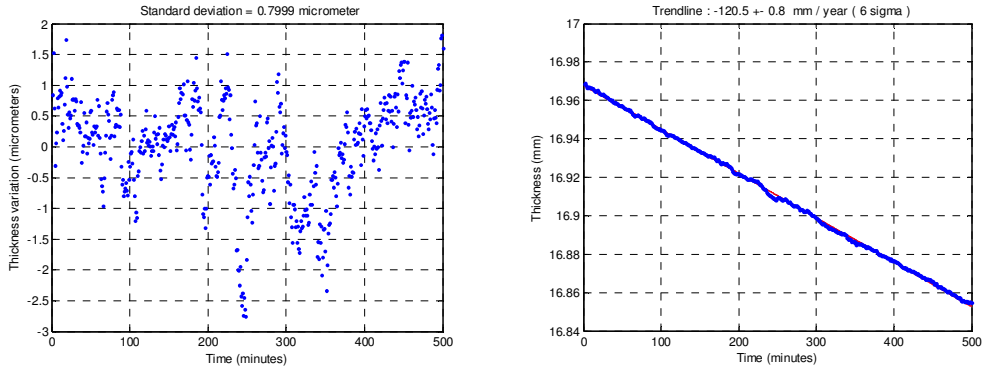
a) b)

Figure 46 Corrosion rate in a) and standard deviation from the trend-line in b).

The variations between the results in Figure 22 and Figure 46 are neglectable.

Parameter set 2:

- Corrosion type: High-rate
- Method for calculating thickness: Correlation

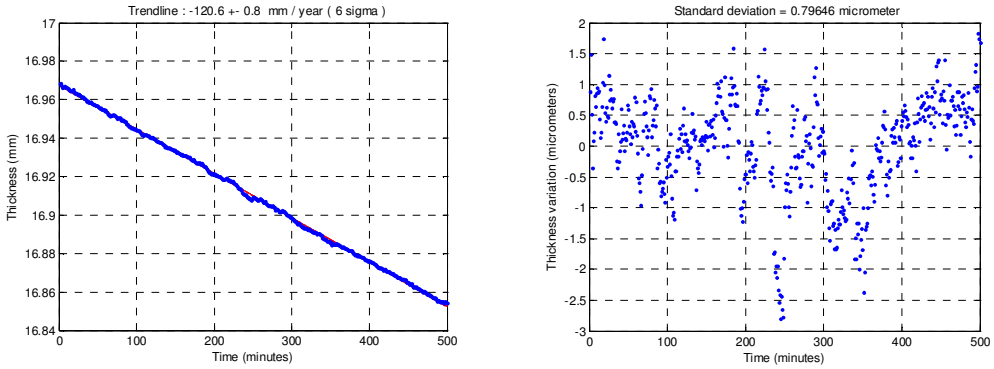


a) b)

Figure 47 Corrosion rate in a) and standard deviation from the trend-line in b).

Parameter set 3:

- Filter type: 7/9
- Corrosion type: High-rate
- Method for calculating thickness: Correlation



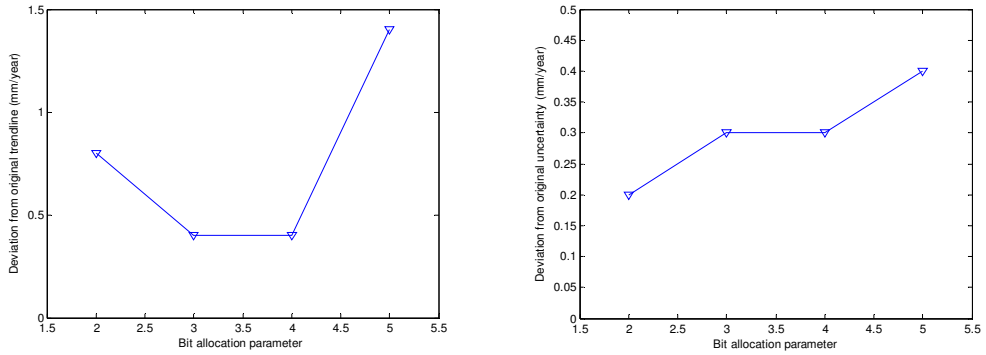
a) b)

Figure 48 Corrosion rate in a) and standard deviation from the trend-line in b).

Optimizing bit-allocation parameter – high-rate

When optimizing this parameter, all the other parameters are kept constant. The optimization is performed by comparing the trendline plots between the unprocessed traces and the compressed traces. From earlier work [32] one can assume that this parameter is 4 or less. Recall that dividing the standard deviation with 2, as is the case in [32], is equivalent to divide the variance by $2^2 = 4$. The following parameters are chosen:

- Filter type: 7/9
- Rate budget: 3500 bits
- Corrosion type: High-rate
- Method for calculating thickness: Inversion
- Dead-zone parameter: 3



a) b)

Figure 49 Deviation from original trendline in a) and deviation from original uncertainty in b) with respect to the bit allocation parameter.

By examining Figure 49 it is clear that all the bit allocation parameters tested have good performance, but one should not exceed 4. 4 is the chosen value in this thesis.

In these tests, the bit ratio is in the range from 1.5 to 1.8 bits/sample, where the ratio increases with a decreasing bit allocation parameter. Recall that the run-length encoding is not included in the bit allocation scheme so even though the rate budget of 3500 bits corresponds to 1.7 bits/sample, the final bitrate may deviate from that.

Thus we may conclude that a larger bit allocation parameter makes the run-length encoder perform better.

Optimizing bit-allocation parameter – low-rate

When optimizing this parameter, all the other parameters are kept constant. The optimization is performed by comparing the trendline plots between the unprocessed traces (Figure 26) and the compressed traces. Two different parameter sets are chosen:

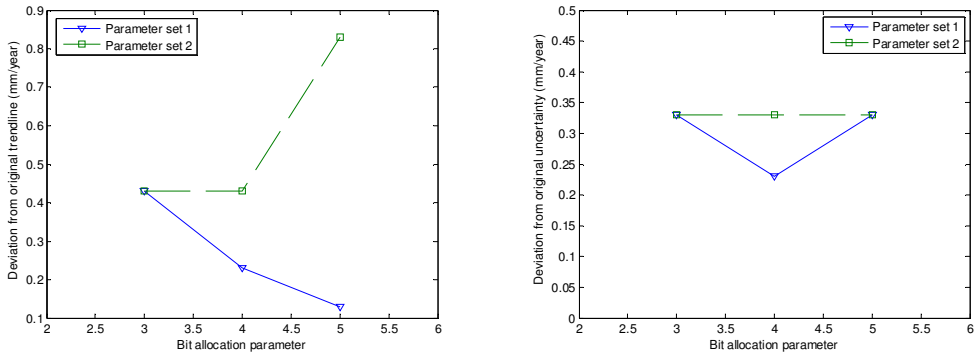
Parameter set 1:

- Filter type: 10/10
- Rate budget: 4000 bits
- Corrosion type: Low-rate
- Method for calculating thickness: Inversion
- Dead-zone parameter: 2

Parameter set 2:

- Filter type: 7/9
- Rate budget: 3500 bits
- Corrosion type: Low-rate
- Method for calculating thickness: Inversion
- Dead-zone parameter: 3

Note that *parameter set 2* equals the parameters chosen in the equivalent high-rate optimization case.



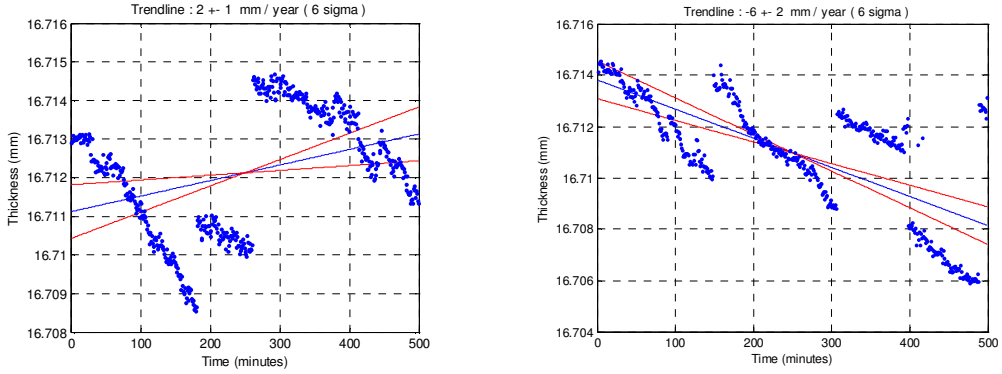
a) b)

Figure 50 Deviation from original trendline in a) and deviation from original uncertainty in b) with respect to the bit allocation parameter.

The minimal deviation from the original uncertainty in Figure 50 b) is 0.23 mm/year which corresponds to an uncertainty of 0.3 mm/year (the uncertainty of the original trendline in Figure 26 is +-0.07 mm/year, and adding this to +-0.23 mm/year gives +-0.3 mm/year.) Thus the uncertainty has increased with a factor 4 compared to the unprocessed traces, giving a relative uncertainty of 18.8%.

Figure 50 shows that the optimal bit allocation parameter equals 4 when considering both sets of parameters. Recall that this conforms to the optimal value with respect to the SNR. As usual we put most emphasis on the deviation from the original uncertainty.

When the bit allocation parameter equaled 2, the term "linear trendline" more or less lost its meaning. It's clear that such results, which are shown in the figure below, *cannot* be trusted. In Figure 51 a) the trendline is actually positive! This indicates that the low-rate experiment is very sensitive to the choice of parameters.



a) b)

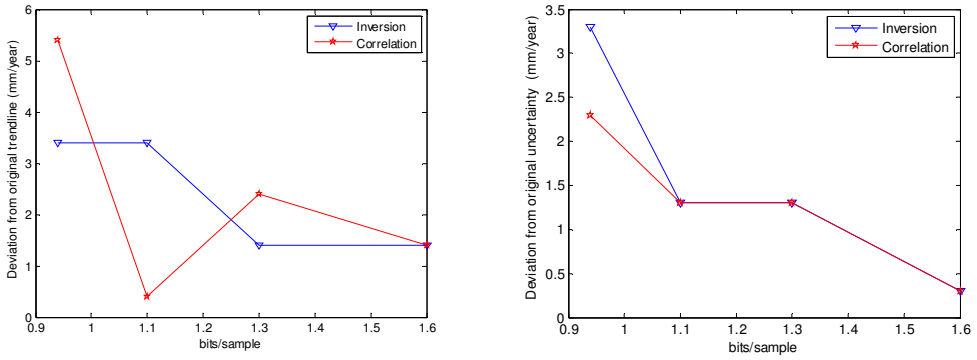
Figure 51 The trendlines cannot be trusted when the bit allocation parameter equals 2. Parameter set 1 is used in a) while parameter set 2 is used in b).

A possible explanation of why a larger bit allocation parameter yields better result in the low-rate case, is that more bits are allocated to the high-frequency subbands. Recall that the high frequency bands contain the detail-coefficients. This information is probably more important when dealing with such sensitive data than it is in the high-rate case.

Inversion vs. correlation

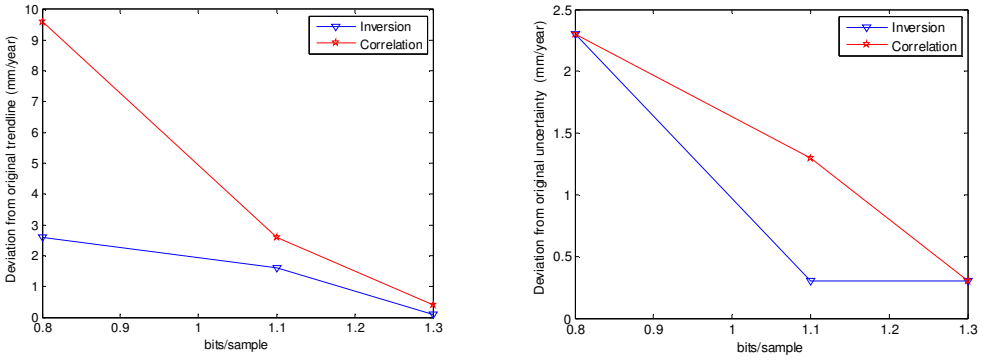
This section compares the trendlines obtained from the inversion method vs. the correlation method in order to see which is the most robust with respect to the encoding scheme. The following parameters are chosen:

- Corrosion type: High-rate
- Dead-zone parameter: 2
- Bit allocation scheme: BWP-algorithm



a) b)

Figure 52 Deviation from original trendline in a) and deviation from original uncertainty in b) with respect to bitrate. In this scenario the 7/9-filter is applied.



a) b)

Figure 53 Deviation from original trendline in a) and deviation from original uncertainty in b) with respect to bitrate. In this scenario the 10/10-filter is applied.

The above figures illustrate that when the 10/10-filter is applied, the inversion method is better for all the tested bitrates. The picture is more unclear when the 7/9-filter is used though. Then the two methods perform about equal.

All in all, based on the above results, the inversion method performs best. Therefore this will subsequently be the preferred method for thickness calculations.

Stair-case effect for low bitrates (high corrosion-rate experiment)

For low bitrates, the fine, linear thickness development seen in Figure 22 is replaced with "stair-case"-like results seen in the below figure. The bitrate is about 0.7 bits/sample. These "steps" are typical when the uncertainty for the trendline is +-3 mm/year or above.

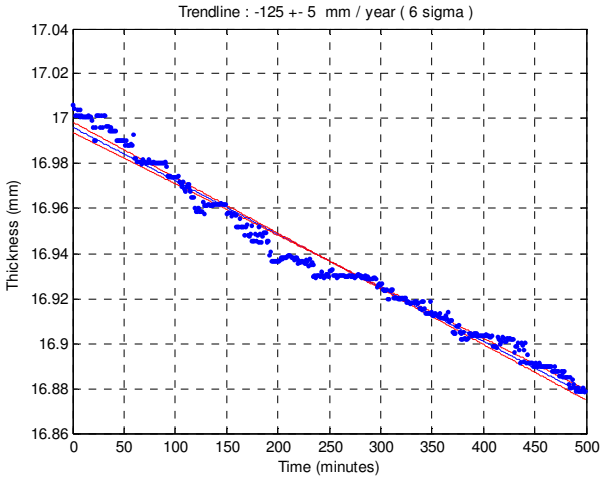


Figure 54 Stair-case effect: The calculated thicknesses seem to create steps.

The reason why this effect occurs is that two subsequent ultrasonic traces have very little differences. For high compression ratios these already small differences may vanish during the encoding process. But then of course, the sum of many small differences results after a while in a big enough change so that a new representation in the transform domain after quantization occurs. This is what causes the discontinuities which look like steps. These results apply for the high-rate traces in particular.

Downsampling and aliasing – low-rate traces

If one downsamples the 500 traces with a factor 2 without using a LP-filter and then interpolates with 2, one obtains the following trendline:

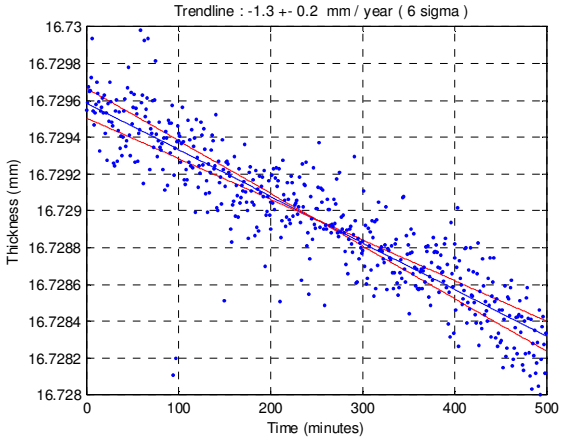


Figure 55 Trendline for downsampled low-rate traces

The uncertainty has increased from 0.07 mm/year to 0.2 mm/year when comparing with the original trendline in Figure 26 a). This is due to aliasing effects. Hence one should use an anti-aliasing filter if the sampling frequency is reduced, or if the incoming signal is downsampled.