NTNU
Innovation and Creativity

Distributed source coding in sensor networks

A practical implementation

Sigmund Seehuus Petersen

Master of Science in Electronics
Submission date: January 2007
Supervisor: Tor Audun Ramstad, IET

Norwegian University of Science and Technology
Department of Electronics and Telecommunications

Problem Description

The goal of the work is to find the best practical implementation with respect to compression and coding of the data to be transmitted by sensors in a wireless network. This is desirable because compression will help reaching the tight requirements on transmitted effect in each sensor. The recently developed scheme of distributed source coding is a revolutionary way of doing this.

Assignment given: 04. September 2006
Supervisor: Tor Audun Ramstad, IET

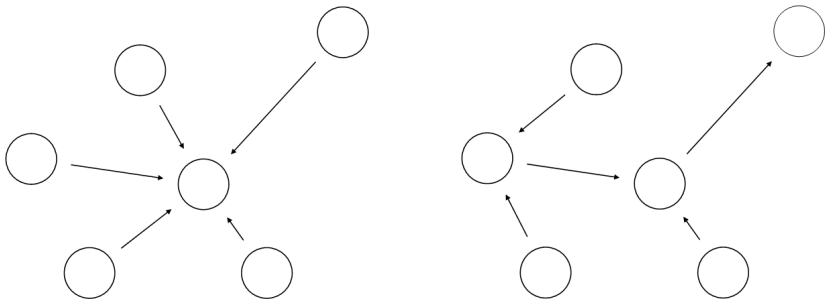
Contents

1 Introduction to Sensor Networks

Wireless sensor networks is an emerging technology with great potential. The main idea is to deploy small, energy-efficient sensor nodes in a chosen area of interest. The scheme can be adopted to and implemented in a vast number of areas. This could be military surveillance, traffic control and habitat monitoring among others. The reason why sensors should be wireless, and not with wired connections, is ease of deployment and the ability to construct dynamic networks, networks which is open for change in topology. Developments in battery technology make power supplies unnecessary. Tiny chips with sensing and communication abilities may last several years, or how long one desires if run on solar energy.

The sensor circuits need to be equipped with transmit (and in some cases receive) functionality to deliver their message. Traditionally, information theory focuses on designing the decoder with low complexity and putting most of the computations on the encoder side. This is not the case in wireless sensor networks. Because the individual sensors depend on limited power supplies, they need to turn their focus to low energy consumption. As a consequence, the main complexity burden needs to be moved from encoder to decoder. This creates the need for a low-complexity encoding algorithm.

The network can be designed in various ways. We usually divide the alternatives in two groups; the fusion based and the ad hoc based design. The main difference between the two is the following. In the ad hoc scheme each node needs to be equipped with both transmitting and receiving capabilities, while with a fusion center the nodes only have to transmit their data. This is illustrated in Fig. 1.



(a) Fusion (b) Ad Hoc

Figure 1: WSN with different topologies

In this thesis we will focus on the challenge of increasing the energy-efficiency. One way to do this is to find better methods for local compression at the sensor nodes by utilizing correlation in time. In addition, in most cases there is considerable correlation between the different sensor data. We can use this redundance to reduce the amount of data each sensor needs to transmit. This can be done in two ways, either by letting the sensors communicate with each other or, more challenging, without any communacation at all. The latter scenario is what we call *distributed source coding*. We take a closer look at this in the next section.

Several research projects have been launched to find the best technologies, strategies and

protocols for sensors and sensor networks. Smart Dust [1, 2] was a leading project that investigated the possibilty for hundreds or thousands of sensor nodes of cubic-millimeter dimension scattered about an environment of interest. A spin off of the project resulted in the company Dust Inc. that aims to deliver tiny sensor chips in near future. More and more providers offer sensor *motes* for commercial application and we will see a growing amount of practical employments of these in the years to follow. A group of animal life researchers deployed a network of wireless sensors on an isolated island to measure changes in the living conditions for birds [3], and this type of habitat monitoring will be important for environment and animal preservation.

As a concluding remark we quote from [4]: "The flexibility, fault tolerance, high sensing fidelity, low cost and rapid deployment characteristics of sensor networks create many new and exciting application areas for remote sensing. In the future, this wide range of application areas will make sensor networks an integral part of our lives."

2 Distributed Source Coding

Distributed source coding is possible when there is correlation among a set of sources. This is exactly the case in typical sensor networks, where the correlation often is high between neighbouring nodes. Each individual node is compressing their data based on observations done by other sensors and not only on its own localized data, hence the term *distributed*. To exploit this correlation and remove the redundance, each node has to know something about what the other sensors send.

This can be done in two ways. First, the senors can communicate with each other throught an intersensor network, or second, they could avoid doing this. The first option gives an extra unwanted overload of creating an additional network and require more processing in each node. The point of source coding in sensor network is exactly the opposite; to reduce the processing amount and thus the energy consumption. This could be avoided using the second scheme. Now the first thing that comes into mind with this option is that without the notion of the other sensors' data, how can the active sensor compress anything at all? Well, as Slepian and Wolf showed in [5] the data can be compressed as much as with knowledge of what the other sensors send. This is known as the Slepian-Wolf theorem and equality is of course only applicable in theory. It is derived asymptotically and based on *random binning* principles. But it may work as a practical goal and give us good measures on how well we are doing.

Imagine you have two sensor nodes and you want to compress the data they transmit as much as possible within a fidelity criterion. Sensor data sequence X is input into the encoder which compresses X based on the correlation distribution between X and Y. Y is sent uncompressed to the decoder as side information as illustrated in Fig. 2. Now the purpose of the joint decoder is to estimate X based on the received data and the side information Y. The more correlated the sources are, the more sure we can be that the chosen estimate is the correct one, or the more we can compress X.

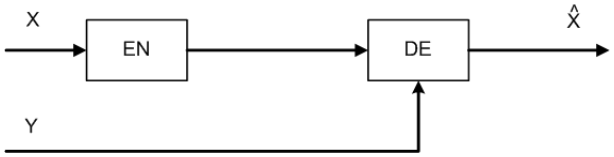


Figure 2: Distributed source coding with side information at the decoder

If one wishes to decode the data losslessly, according to classical source coding theory one may encode with a rate $R_X \geq H(X)$ and $R_Y \geq H(Y)$ for source X and Y respectively. By taking into consideration any correlation between X and Y one is able to encode both sources to their joint entropy $R_X + R_Y \geq H(X,Y)$. In other words, since $H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y)$, it is possible to encode one of the sources to its respective *conditional* entropy $R_X \geq H(X|Y)$ or $R_Y \geq H(Y|X)$ leaving the other at its entropy. Slepian and Wolf claims that this can be done just as well without intersensor communication as with it, gaining $H(X) - H(X|Y)$ or $H(Y) - H(Y|X)$ on the rate. One can easily see that this degenerated to the classical lossless encoding limit ($H(X)$) when there is no correlation (i.e. $H(X|Y) = 0$). We take a closer look at Slepian-Wolf coding principles in Section 3

Now the lossless encoding theory of [5] can only be used on discrete sources. Wyner and Ziv [6, 7] took this further by considering continuous valued sources, thus introducing distortion to the signal to get finite entropies. Now this is often the case in typical sendsor network scenarios. The main thought was to introduce a quantizer step before Slepian-Wolf coding, analogous to the quantizer followed by entropy coding in single-source compression. The quantizer step is a part of the rate distortion theory and can be done in various ways depending on the distribution and memory of the input. Rate distortion theory is presented in Section 4. Wyner-Ziv coding is described in Section 5.

The correlation between the sources may be modelled as a virtual *correlation* channel where X is the input and Y the output of the channel (Fig. 3). The channel is described by the error probability ρ, the probability that Y is different from X. Thus a low ρ gives high correlation which makes it possible to encode with lower rate. Because of this model it is suggested in the literature to use channel coding principles in the source coding. This was first suggested by Wyner in [8] and is thus called *Wyner's scheme*. Pradhan and Ramchandran has done considerable work in this area [9, 10, 11, 12] and a lot of the things described in this thesis are based on this scheme. In the same way that channel coding expands the rate to protect the signal from channel noise, it can be used in the opposite way to reduce the rate in a controlled way. The technique is based on so-called *binning* where all possible outcomes of an input signal are placed in disjoint cosets or *bins*. The theory is called *distributed source coding using syndromes* (DISCUS) and we take closer look an this in Section 6

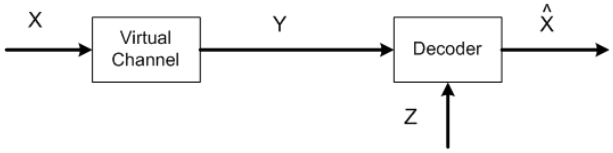


Figure 3: Virtual correlation channel between X and Y

Here we have used the simplification of an asymmetric scheme, i.e. one of the sensors send their data in an uncompressed format. The optimal operation would be to use a symmetric scheme where both sensors send compressed data based on each other's correlation, or even an adaptive symmetric scheme where the nodes vary the transmission rate depending on factors such as channel quality and characteristics of the signal. This is described in Section 7.

Distributed source coding using syndromes may be implemented in several ways depending on the channel coding technique you chose. The most common techniques used are linear block codes, convolutional codes and concatenated codes. Research on channel coding has led to two leading techniques, low density parity-check (LDPC) codes and turbo codes. LDPC codes is a form of linear block codes using low density parity.check matrices. Turbo codes is a concatenated code combining an *interleaving* with convolutional codes. Turbo coding used in distributed source coding is investigated in e.g. [13, 14, 15, 16]. In this thesis we have chosen to follow the linear block coding scheme in its most advanced form of LDPC codes. We have chosen this because of its proven performance over turbo codes and because of its portability to the distributed scheme. LDPC codes are described generally in Section 8 and specially for DISCUS in Section 9.

3 Lossless DSC

Slepian and Wolf described a theorem for compression of a source exploiting the correlation with another source. This was called distributed source coding. In [5] this is done without distortion of the source. The main idea was that a source X should be able to compress its data based on the correlation distribution with a secondary source Y without the need for intersensor communication. Slepian and Wolf showed that it is possible to encode with the same rate no matter if you have the knowledge of what the other encoder sent or not. This theory is compressed to the following formulas [5]

$$R_1 \geq H(X|Y), \tag{1a}$$
$$R_2 \geq H(Y|X), \tag{1b}$$
$$R_1 + R_2 \geq H(X,Y), \tag{1c}$$

and can be depicted as in Fig. 4.

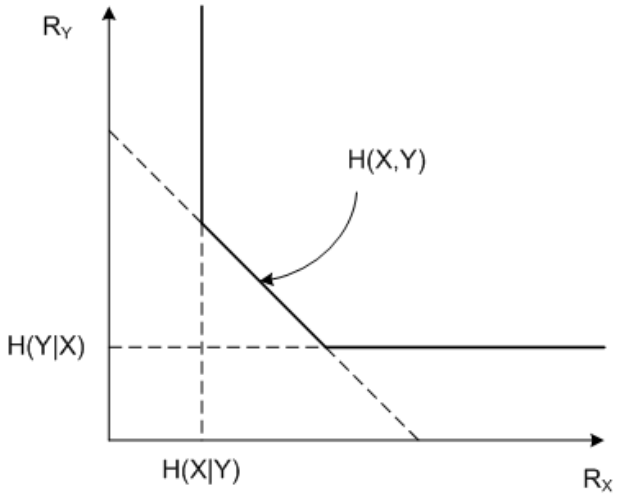


Figure 4: Rate region for two sources

The corner points may be reached by the *asymmetric* coding scheme. All the points on the line $H(X,Y)$ is obtainable by either *time sharing* or by *symmetric* coding. Time sharing is done by letting one of the sources act as side information a part of the time and the other source the rest.

Slepian and Wolf showed this for two discrete variables with finite alpfabet. Later Cover [17] expanded this to arbitrary ergodic processes, countably infinite alphabets and an arbitrary number of correlated sources. In real life discrete processes are not very common, but an illustrative example is given next.

3.1 Slepian-Wolf coding of two binary sources

Say we have two correlated 3-bit binary sources X and Y. The correlation between them is such that the maximum Hamming distance is $d_H \leq 1$. We are now going to code these in a way that enables us to decode them without distortion. Using classic source coding theory it is possible to compress the sources to their respective entropies, $H(X)$ and $H(Y)$ (3 bits here). Taking into consideration that there exists a certain dependency between X and Y we can achcieve savings on the amount of transmitted bits.

Looking at the case where Y is available at the decoder, there is no point in differentiate between $X = 000$ and $X = 111$, since we know that Y is maximum 1 away from X in a Hamming distance sense. Hence we can put all possible outcomes of X into disjoint *bins* or *cosets*. By performing a clever *binning* it is possible to estimate which of the codewords in the respective coset was the original value of X. In the same fashion we can construct the following cosets using codewords of X: {100,111}, {010,101} and {001,110}. By transmitting only the index of the active coset we reduce the amount of transmitted bits, in this case from 3 to 2 bits.

The binning scheme is illustrated in Fig. 5.

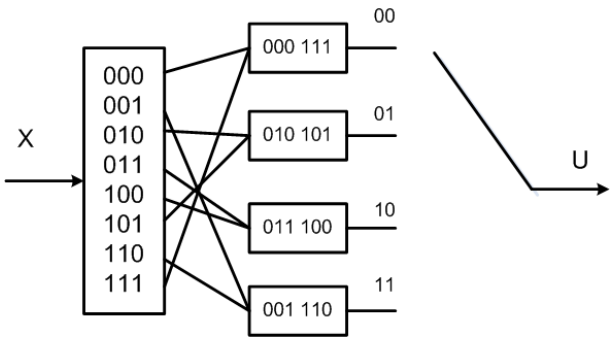


Figure 5: Coset construction

4 Rate-Distortion Theory

When we consider analogue sources, it is no longer possible to decode these losslessly because of the limited capacity of a realistic channel. Instead of having the uncontrolled distortion from the channel we can introduce a certain controlled degradation or distortion to the signal before transmission. The more noise on the available channel, the more we have to increase this distortion, or in other words: reduce the source rate. A simple but intuitive illustration of this result is shown in Fig. 6.

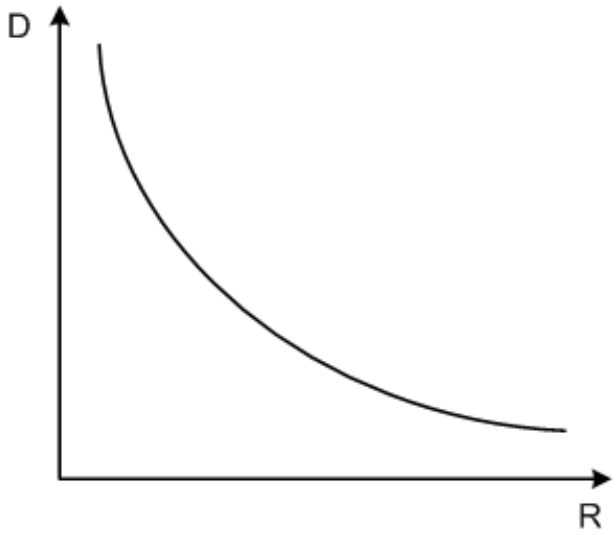


Figure 6: Typical picture of the rate distortion function

The rate distortion function gives us theoretical limits for how well we can do lossy compression with general distortion metrics and general sources. The general rate distortion function with expected distortion

$$\bar{d} \leq D$$

is given by

$$\begin{aligned} R(D) &= \min_{\{y_j\}_{j=1}^{J} \in D} \{R\} && \text{(2a)} \\ &= \min_{\{\text{every } D\text{-allowed channel}\}} I(\mathbf{p};\mathbf{P}) && \text{(2b)} \\ &= \min_{\left\{\mathbf{P}|\bar{d}\leq D\right\}} I(\mathbf{p};\mathbf{P}) \quad \text{bits/source symbol,} && \text{(2c)} \end{aligned}$$

where y_j are the output signals, $\mathbf{p} = p(x_n, y_m)$ and $\mathbf{P}$ is the channel matrix.

The rate distortion function is hard to find for general sources and distortion metrics, but may be specified for special cases. For a gaussian memoryless source with expected quadratic distortion metric $\sigma_N^2 = D$ and input power σ_X^2 the rate distortion function is

$$R(D) = \tfrac{1}{2} \log \left(\frac{\sigma_X^2}{D} \right). \tag{3}$$

From this you can solve and get the distortion rate function

$$D(R) = \sigma_X^2 \cdot 2^{-2R}. \tag{4}$$

The choice of rate is thus a trade-off on how much distortion one accept into the signal or given by certain restrictions (e.g. power limitations, channel noise).

The rate distortion theory described is utilized in practical examples through different forms of quantization, some more sophisticated than others.

4.1 Scalar Quantization

Scalar quantization is done on one sample of a source at a time. This can either be uniform or optimized for the distribution of the source. One way to optimize is by the generalized Lloyd-Max algorithm [18]. The optimized quantizer design is obtained by finding the *minimal quantization error variance* solved by the conditions

$$\frac{\partial \sigma_\epsilon^2}{\partial r_n} = 0 \quad \text{for} \quad n = 0, \ldots, N-1, \quad \frac{\partial \sigma_\epsilon^2}{\partial d_n} = 0 \quad \text{for} \quad n = 1, \ldots, N-1 \tag{5}$$

where r_n are the representation levels, d_n are the decision levels and N the number of levels in the quantizer. These conditions can be solved into a couple of equations

$$d_{n,opt} = \tfrac{1}{2}(r_{n,opt} + r_{n-1,opt}) \quad \text{for} \quad n = 1, \ldots, N-1 \tag{6a}$$

$$r_{n,opt} = \frac{\int_{d_{n,opt}}^{d_{n+1,opt}} x p_x(x) dx}{\int_{d_{n,opt}}^{d_{n+1,opt}} p_x(x) dx} \quad \text{for} \quad n = 0, \ldots, N-1 \tag{6b}$$

which have to solved numerically by iteration from some initial values. Equation 6 shows that in the optimized quantizer a given representation level is the *centroid* of the corresponding decision interval, and the decision levels lies *in the middle* of two representation levels. The quantizer characteristics will have large decision intervals where the probability of occurence of the input signal is small, and vice versa. This way each quantization level will contribute with the same amount to the minimized quantization error. [19]

4.2 Vector Quantization

Vector quantization is done, as the name implies, on a vector of samples. When the quantization is done on more than one sample, the decision levels become multidimensional, or *regions*, and the signal is quantized to the representation point of a region. The representation level is decided based on N consecutive values of the input signal. If the regions

are *Voronoi* regions (or *Dirichlet* partitions) and the representation levels are centroids of their regions, the quantizer is optimized.

The *representation vectors* form the *codebook* of a vector quantizer (VQ). The index of the active codeword is transmitted over the channel and hopefully reconstructed on the receiver side. The codebook is often designed by *training* of the VQ. That is, you input a known sequence of bits that you assume *looks like* a potential source the VQ will be used on, and you find the codebook by iterative algorithms.

The more correlated in time a source is, the better the VQ performs. Vector quantization can be shown to give the *best rate distortion performance* for a given degree of compression. However, due to the high dimensionality and large codebook size, it leads to high computational complexity and delay and is not very practical. [19, 20]

4.3 Nested Quantization

In nested quantization you have two quantizers with different rates nested into each other. We say that there is a *fine* code and a *coarse* code. The coarse code can be seen as a coset construction step; you first quantize using the fine code, then you use the coarse code to place your codeword in a coset.

The nested quantization scheme may be done on any blocklength of the source samples, i.e it may be constructed in any dimension. If both codes are in two dimensions we call it *nested lattice quantization*. This becomes the same as vector quantization, and optimization of the *Voronoi regions* is a crucial design feature. For the two-dimensional case the optimal structure is found to be the hexagonal lattice [21].

5 Lossy DSC

Wyner-Ziv coding is Slepian-Wolf Coding with a distortion measure, i.e it is lossy compression taking a correlation distribution into consideration. Lossy source coding is necessary when you do not have a channel with infinite capacity at your disposal (or if you want to be able to decode your signal within a fidelity criterion). Wyner-Ziv coding can be illustrated as quantization followed by lossless Slepian-Wolf coding as in Fig. 7.

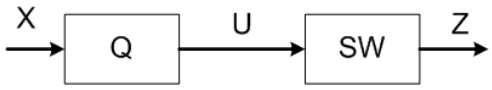


Figure 7: Block diagram of Wyner-Ziv Coding

The Wyner-Ziv rate region is not specified for general sources, but the important special cases of binary and gaussian input are well known in the literature. The general Wyner-Ziv rate region can be illustrated as in Fig. 8 with an inner and outer bound. The achievable rates lie between these two bounds. [22]

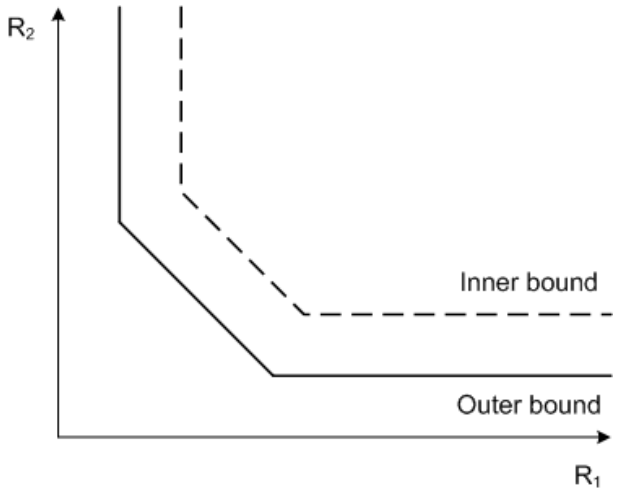


Figure 8: The inner and outer rate regions in lossy DSC

Since the Wyner-Ziv theory is a combination of Slepian-Wolf coding and rate-distrotion theory described Sections 3 and 4, we will focus here on some important practical examples.

5.1 The binary symmetric case

X and Y are two binary sources and the correlation between them is modelled as a binary symmetric channel with error probability ρ with Hamming distance as the distortion measure. If we write $X = Y \oplus E$, where E is a Bernoulli(ρ) source, the rate distortion function with Y known at the encoder *and* the decoder given as

$$R_{X|Y}(D) = R_E(D) = \begin{cases} H(\rho) - H(D), & 0 \leq D \leq min\{\rho, 1-\rho\}, \\ 0, & D \geq min\{\rho, 1-\rho\}. \end{cases} \tag{7}$$

If however Y only is known at the decoder, then the *Wyner-Ziv* rate distortion function is given as

$$R^*_{WZ}(D) = l.c.e\{H(\rho * D) - H(D), (\rho, 0)\}, \quad 0 \leq D \leq \rho, \tag{8}$$

thw lowest convex envelope (l.c.e) of $H(\rho * D) - H(D)$ and the point $(D = \rho, R = 0)$, hvor $\rho * D = (1-\rho) * D + (1-D)\rho$. For $\rho \leq 0.5$, $R^*_{WZ}(D) \geq R_{X|Y}(D)$ with equality only in two points: the zero-rate point $(\rho, 0)$ and the zero-distortion point $(0, H(\rho))$. Thus Wyner-Ziv coding suffers rate loss in the binary symmetric case. When $D = 0$, the Wyner-Ziv problem degenerates to the Slepian-Wolf problem with $R *_{WZ} (0) = R_{X|Y}(0) = H(X|Y) = H(\rho)$.

5.2 The quadratic gaussian case

In this case we have two gaussian stochastisc variables X_k and Y_k with variance σ_x^2 and σ_y^2 and correlation coefficient ρ (Note: greater ρ gives more correlated sources in this case), and we let $\mathbf{D} = (D_x, D_y)$ be the distortion criteria. We then have that if [23]

$$d_x \leq \frac{D_x}{\sigma_x^2}, d_y \leq \frac{D_y}{\sigma_y^2}$$

then we get the conditions

$$R_X \geq \tfrac{1}{2} \log \left[\frac{(1-\rho^2)\beta}{(1-\rho^2)\beta d_y - 2\rho^2 d_x d_y} \right], \tag{9a}$$

$$R_Y \geq \tfrac{1}{2} \log \left[\frac{(1-\rho^2)\beta}{(1-\rho^2)\beta d_x - 2\rho^2 d_x d_y} \right], \tag{9b}$$

$$R_X + R_Y \geq \tfrac{1}{2} \log \left[\frac{(1-\rho^2)\beta}{2 d_x d_y} \right] \tag{9c}$$

where

$$\beta = 1 + \sqrt{1 + 4\rho^2 d_x d_y (1-\rho^2)^{-2}}.$$

We get minimal rate by setting $d_x = D_x/\sigma_x^2$ og $d_y = D_y/\sigma_y^2$ which gives

$$\beta_{max} = \beta(\frac{D_x}{\sigma_x^2} \frac{D_y}{\sigma_y^2})$$

and

$$R_X + R_Y = \tfrac{1}{2}\log^+\left[(1-\rho^2)\frac{\beta_{max}}{2}\cdot\frac{\sigma_x^2}{D_x}\frac{\sigma_y^2}{D_y}\right]. \tag{10}$$

According to classic rate distortion theory we may encode two memoryless gaussian sources to

$$R_X + R_Y = \tfrac{1}{2}\log\left[\frac{\sigma_X^2\sigma_Y^2}{D_xD_y}\right]. \tag{11}$$

Taking into consideration the correlation we achieve a coding gain of

$$\Delta R = \tfrac{1}{2}\log\left[\frac{(1-\rho^2)\beta max}{2}\right]. \tag{12}$$

Observe that $\Delta R = 0$ when $\rho = 0$ and that $\Delta R \to \frac{1}{4}\log(\sigma_x^2\sigma_y^2/D_xD_y)$ when $\rho \to 1$.

6 Distributed source coding using syndromes

In the distributed compression setting, as in the non-distributed one, the continuous samples need to be quantized in order to obtain a finite entropy.

As mentioned before, the correlation between the sources in a sensor network may be modeled as a "correlation" channel. The idea is then to use a channel coding approach to compress the source data even further. The quantized codeword U is of course correlated to X. If X is correlated to the side information Y, U is also correlated to Y. So the *correlation channel* can be described by the conditional distribution $P(Y|U)$. The side information carries the information $I(U;Y)$ about U which can be exploited on the decoder side to estimate X. Now the correlation distribution may differ considerably from case to case, but it is often modeled in literature as a *Binary Symmetric Channel* (BSC) or a channel with *Additive White Gaussian Noise* (AWGN).

So how do we reduce the amount of bits representing X without knowing exactly what the corresponding sample of Y is? Let us illustrate with an example. Let X and Y be two equiprobable 3-bit words where the correlation is given by a Hamming distance not more than one. If Y was known both at the decoder *and* at the encoder, there would be no point in representing X with more than 2 bits. (Given Y, $X \oplus Y$ is in the set $\{000, 001, 010, 100\}$ where $\oplus$ is the modulo-two sum). With Y known *only* at the decoder, is this still possible? Yes, there is no point sending both $X = 000$ and $X = 111$ because the Hamming distance between them is 3. With Y available, one of the codewords in the set is uniquely chosen. Now all the possible codeword representations of X can be sorted into similar sets giving the additional sets of X: $\{001, 110\}, \{010, 101\}$ and $\{100, 011\}$. Hence X needs only be transmitted with 2 bits instead of 3.

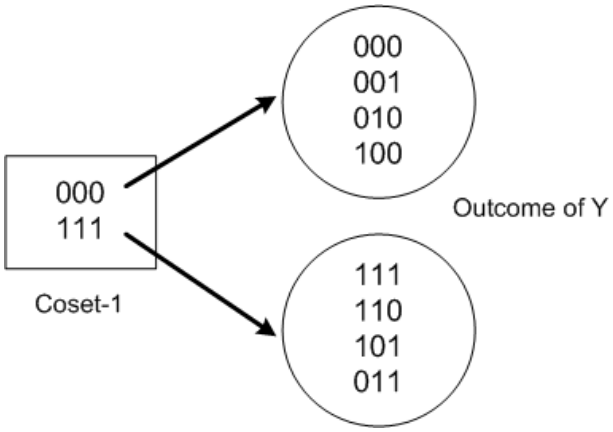


Figure 9: Possible outcomes of Y if X lies in the coset $\{000, 111\}$

This is a primitive example but it illustrates the idea of the distributed source coding scheme. The goal is to find a good channel code that performs close to the Wyner-Ziv limit from section 5. Observe that the first set of two codewords of X is a $(3, 1, 3)$ linear block code, also known as a 3-bit repetition code. The other sets are just variants or *cosets* of this repetition code. Thus, instead of describing X by its 3-bit value, we encode *which coset X belongs to*, incurring a cost of 2 bits, just as in the case when Y is known

both to the encoder and the decoder. Now recall that the linear block code can be given by its *parity-check matrix* H. Every coset of a linear code is associated with a unique *syndrome* with $s = H^T c$, where c is any valid codeword.

Promising channel coding techniques that have been mentioned as alternatives are turbo coding and LDPC coding. Both have been described and tested by different researchers. We have chosen the LDPC scheme in this thesis because it has proven the best transition to distributed compression. This coding technique will be discussed generally in section 8 and specifically for DSC in section 9.

7 Symmetric design

The asymmetric case of distributed source coding is not very flexible in rate allocation. If we want to vary the rate of the different sources in the asymmetric case we need to use time sharing which introduces a synchronization issue; the sensors need to communicate to synchronize. This is exactly what we were trying to avoid in the first place. Two different solutions have been suggested to this problem. In [9] they propose a method base on the same principles as in the DISCUS scheme, but with a more advanced coding and decoding. Another way of doing it is by *source splitting*. This is discussed in e.g. [24], but will not be described further here.

We will focus on the method introduced in [9] and further developed in [12]. Unlike the asymmetric case where one source sends its information losslessly and the other source sends compressed data, in the symmetric case both sources should be able to send only partial information without compromising the reconstructed signal quality at the decoder. A symmetric encoding scheme should be able to encode with rates in all of the achievable rate region of Fig. 4.

Consider the encoding of two general correlated sources X and Y. We are going to encode them in a symmetrical way, i.e. we want to compress each source with any rate ranging between $H(X)$ and $H(X|Y)$ for X, and between $H(Y)$ and $H(Y|X)$ for Y. Assume that $H(X) \leq H(Y)$. Following the channel coding strategy, we search for two generator matrices G_x and G_y containing $n(1 - H(X|Y))$ and $n(1 - H(Y))$ rows to achieve the corner point $(H(X|Y), H(Y))$ (see Fig. 4). To assign different rates we move some of the rows from G_x to G_y to move along the line $H(X, Y)$ until we reach the other corner point $(H(X), H(Y|X))$.

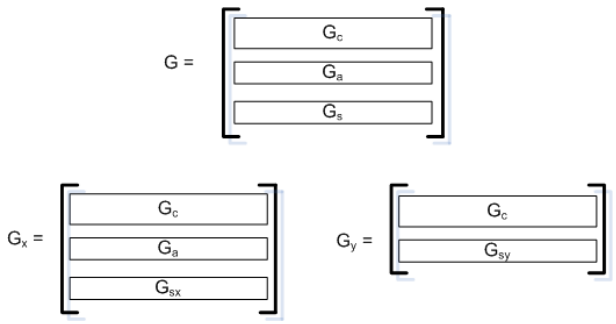


Figure 10: Construction of generator matrices for the symmetric case

Consider a generator matrix $\mathbf{G}_c$ of size $n(1 - H(Y)) \times n$ with linearly independent rows, where n is the block-length used in encoding. This generator matrix can be used to partition the space of n-length Y sequences. Thus $\mathbf{G}_y = \mathbf{G}_c$, and the encoder of Y sends the syndrome associated with $\mathbf{G}_y$. The decoder can now recover n-length Y sequences based on this information and the knowledge of the statistics of Y. To encode X we need to find a generator matrix $\mathbf{G}_x$. Consider a matrix $\mathbf{G}_a$ with $n(H(Y) - H(X)) \times n$ linearly independent rows. A matrix formed by stacking $\mathbf{G}_c$ and $\mathbf{G}_a$ can be used to partition the space of n-length sequences with $nH(X)$ cosets. To reduce the rate induced by this stacked matrix from $H(X)$ bits/sample to $H(X|Y)$ bits/sample we construct a matrix $\mathbf{G}_s$

with $n(H(X) - H(X|Y))$ linearly independent rows. Now $\mathbf{G}_x$ is formed by stacking $\mathbf{G}_c$, $\mathbf{G}_a$ and $\mathbf{G}_s$. The encoder of X sends the syndrome of the observed n-length sequence of X with respect to $\mathbf{G}_x$ to the decoder. The decoder having recovered n-length Y-sequence can now use a standard decoding algorithm to recover the n-length X sequence using the joint distribution $p(x, y)$ of X and Y.

To trade rates between the encoders of X and Y, any number of the rows of $\mathbf{G}_s$ can be moved from $\mathbf{G}_x$ to $\mathbf{G}_y$. Finally when $\mathbf{G}_x$ consists only of $\mathbf{G}_c$ and $\mathbf{G}_a$, the encoders of X and Y will be transmitting at rates $H(X)$ and $H(Y|X)$ bits/sample respectively. This allocation process can be viewed as dividing the generator matrix $\mathbf{G}$ into two ($\mathbf{G}_x$ and $\mathbf{G}_y$) as illustrated in Fig. 10.

7.1 Decoder structure

Remember for the asymmetric case that to decode the sources one need to find the coset where the compressed source (X) exists and find the codeword in that coset that is closest to the side information (Y). Now in the symmetric case both soures send partial information so this decoding strategy is not applicable.

We will come back to this in the specific case of LDPC decoding.

7.2 Multiple sources

A sensor network with two sensors as we have looked at so far is not much of a sensor network to talk about. It is desirable and absolutely necessary to be able to present a theory for more than two sources. The beauty of the ideas presented so far in this thesis is that this is not a challenge. The example of two sources in symmetric distributive source coding is easily extended to multiple sources.

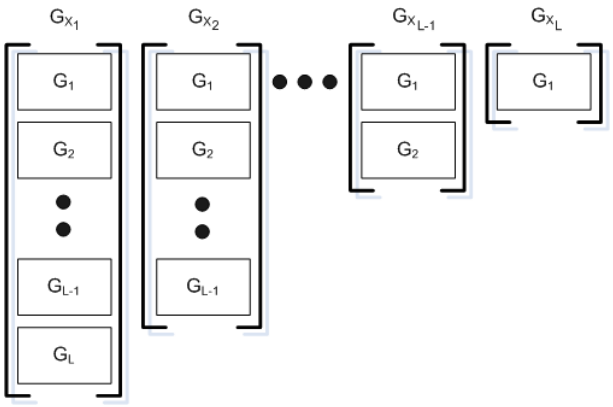


Figure 11: Matrix definition for multiple sources

For L sources, let us form an ordering of some arbitrary sources $\{X_1, \ldots, X_L\}$ such that $H(X_1|X_2, \ldots, X_L) \leq \ldots \leq H(X_i|X_{i+1}, \ldots, X_L) \leq \ldots \leq H(X_L)$. Without loss of generality we may arrange the sources in this fashion. We can achieve the corner point

$H(X_1|X_2,\ldots,X_L),\ldots,H(X_i|X_{i+1},\ldots,X_L),\ldots,H(X_L))$ by recursively defining the generator matrices of each code from a single generator matrix, as shown in Fig. 11.

The process begins with defining the generator matrix $\mathbf{G}_L$ with $n(1-H(X_L))$ linearly independent rows. The other matrices can be found by iterarively defining $\mathbf{G}_{i-1}$ as the stacking of $\mathbf{G}_i$ and a matrix $\mathbf{A}_{i-1}$ with $n(H(X_i|X_{i+1},\ldots,H(X_L))-H(X_{i-1}|X_i,\ldots,X_L))$ linearly independent rows. Clearly, as in the two-source case, the non-corner points can be achieved by trading the specific rows of the generator matrices between each other.

8 LDPC coding

Low-density parity-check coding is a form of linear block codes with iterative decoding. An LDPC code is determined by its parity-check matrix H or equivalently by its bipartite Tanner graph representation. The bipartite graph is used in the message-passing decoding algorithm.

An LDPC code is a binary linear code with an $M \times N$ sparse parity-check matrix $\mathbf{H}$, i.e. $\mathbf{H}$ contains mostly 0's and relatively few 1's. The code may be regular or irregular. A *regular* LDPC code has exactly w_c ones per column and exactly $w_r = w_c(N/M)$ ones per row in $\mathbf{H}$, where w_c and w_r are small compared to N.

Any parity-check code, including the LDPC code, may be specified by a Tanner graph. A Tanner graph is a representation of a code corresponding to a set of parity checks that specify the code. The graph contains two kinds of nodes, check nodes and bit nodes. There are M check nodes, one for each parity check $C_1, ..., C_m, ..., C_M$, and N bit nodes, one for each code bit $v_1, ..., v_m, ..., v_M$. The check nodes are connected to the bit nodes that they check. Specifically, a branch (edge) connects check-node m to bit-node n if and only if the mth parity check involves the nth bit (i.e. only if $\mathbf{H}_{m,n} = 1$). Thus, the graph is analogous to the $\mathbf{H}$ matrix. The mth parity check is regarded as a "local constraint" stating the condition $\sum_{j=1}^{n} h_{ij}x_j = 0$. A given configuration $(x_1, ..., x_n)$ is a valid codeword if and only if all local constraints are satisfied. [25]

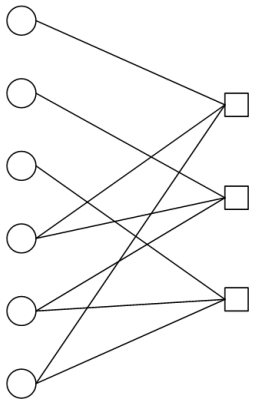


Figure 12: Tanner graph representation of a (6,3) code

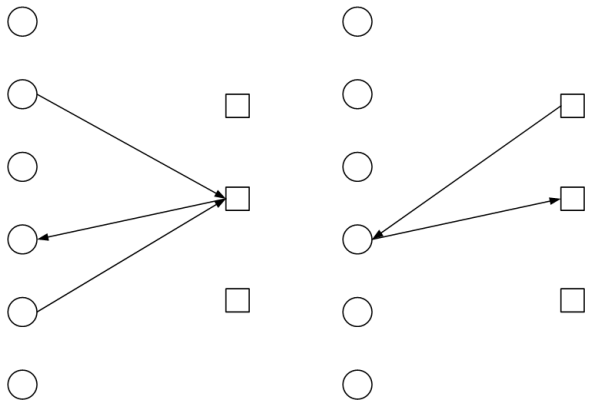
Decoding of LDPC codes

For decoding of LDPC codes, we want to find the probability that each bit v_n of a recieved vector $\mathbf{r}$ equals 1 or 0, knowing that the estimated codeword $\widehat{\mathbf{U}}$ stemming from $\mathbf{r}$ satisfy the constraint $\widehat{\mathbf{U}}\mathbf{H}^{\mathbf{T}} = 0$. Given a received vector $\mathbf{r}$, solving directly for the probability $P(v_n = b|\mathbf{r})$, that the nth bit equals either one or zero is very complex.

Gallager provided an iterative technique, known as the sum-product algorithm, where the probability $\mu_{mn}(b)$ that the mth check is satisfied by a received vector $\mathbf{r}$, is passed from check node C_m to bit node v_n. This satisfied-check probability $\mu_{mn}(b)$ is collected from all bits participating in the mth check other than v_n. Likewise, the bit probability $q_{mn}(b)$ that the nth bit has value $v_n = b$, is passed from bit node v_n to check node C_m. This bit probability $q_{mn}(b)$ is collected from all the checks that the nth bit participates in other than C_m. [26]

We follow the example of the $(6, 3)$ LDPC code from Fig. 12. Messages containing satisfied-check probabilities $\mu_{mn}(b)$ are shown moving from check nodes, and messages containing bit probabilities $q_{mn}(b)$ are shown moving from bit nodes to check nodes. The process is repeated until it converges to a code word solution or until a predefined number

of iterations is reached. We follow the message-passing algorithm focusing particularly on bit-node v_4 and check node C_2. Bit nodes are initialized with likelihood values stemming from a detector. Suppose that bit node v_4 passes the probability $q_{24}(b)$ that $v_4 = 1$ to check-node C_2. Check-node C_2 collects the incoming probabilities from all other bits involved in check 2 (v_2 and v_5), computes a probability $\mu_{24}(b)$ that parity-check C_2 is satisfied given that $v_4 = 1$, and passes this message to bit-node v_4. Check-node C_2 passes similar information to v_2, given $v_2 = 1$, and to v_5, given $v_5 = 1$. When bit-node v_4 receives such satisfied-check information from all the check nodes involving v_4 (C_1 and C_2), it recomputes the probability that $v_4 = 1$ collected from the connected check-nodes apart from C_2 and passes this message back to node C_2, similar information to C_1, and so forth. The process is illustrated in Fig. 13



(a) $\mu_{24}(b)$ (b) $q_{24}(b)$

Figure 13: Sum-product algorithm

9 LDPC coding in the distributed setting

While LDPC coding (or linear block coding in general) in the traditional encoding structure expands the bandwith (increases the rate), in DSC it is used for bandwith compression. Thus we have a dual situation and we can think of it as switching the roles of the encoder and the decoder. To get the desired channel codeword, we multiply the source codeword with the parity check matrix $\mathbf{H}$. (Remember that in channel coding for error protection we multiply with the generator matrix $\mathbf{G}$ where $\mathbf{GH^T} = \mathbf{I}$). This would give us the syndrome of the received codeword in the traditional sense, and in the same fashion gives us now the "syndrome" on the encoder side, i.e. the index we want to transmit over the channel. We will still call the matrix used on the encoder side for the generator matrix $\mathbf{G}$, but keep in mind that in this setting this is the "opposite" of the generator matrix in the conservative use.

9.1 Code construction for the symmetric case

Given $\mathbf{G}$, to encode, i.e. compress, an arbitrary binary input sequence, we multiply $\mathbf{X}$ with $\mathbf{G}$ and find the corresponding syndrome $\mathbf{Z}$ of length $(n-k)$. This is what we transmit over the channel and it represents the index of the coset containing our active codeword. The goal is to recover this using the received bits from the other sensors as we have seen earlier.

For decoding, the decoder must estimate the n-length sequence $\mathbf{X}$ from its $(n-k)$-long syndrome $\mathbf{Z}$ and the corresponding n-length sequence $\mathbf{Y}$. This is done by a modified version of the sum-product algorithm described in section 8. For the corner cases (source coding with side information) we use the factor graph depicted in Fig. 14. This is based on the same structure as the graph in Fig. 12, there is a set of constraints (squares) and a set of variables (circles). In addition we need the received bits from the side information and an extra row of constraints based on the correlation information.

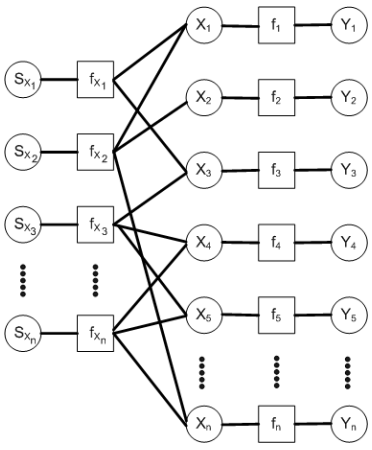


Figure 14: Decoding with the Tanner graph in the single-machine case

In order to achieve any desirable rate in the rate region, we add a line of the compressed variable bits of the side information to the Tanner graph and the corresponding con-

straints belonging to an equivalent single-machine code for the side information (Fig. 15). Decoding is again achieved by the sum-product algorithm on this graph. This expansion of the graph can be done in the same manner to include multiple sources into the symmetric distributed LDPC setting. The additional single-machine codes is connected to the graph through the correlation constraint row in the middle of the bipartite graph.

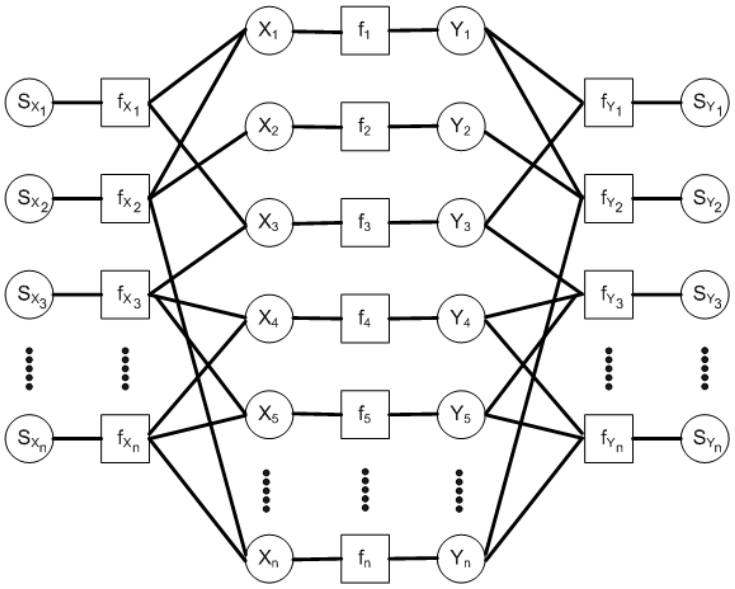


Figure 15: Decoding with the Tanner graph in the two-machine case

10 Implementation

To test the theory and ideas presented in this thesis a distributed source coder using LDPC codes is implemented in C/C++. In the implementation we have used an enhanced version of the LDPC code of MacKay and Neal in C. The system is simulated and tested in both the C/C++ application and in Matlab. Due to lack of time the system is not complete and should be continued in future studies. The source code can be found as an appendix to this thesis.

The goal of the implementation was to construct a distributed source coder an use it on a set of ECG data using the described LDPC coding scheme. ECG is a kind of medical data and makes a nice input to an example for medical wireless sensor networks, an interesting area of application of DSC. The data are generated in two (or more) "sensors" with a known spatial correlation distribution. This correlation is in the form of a gaussian distributed noise figure N with zero mean and known variance. The modelled correlation varies with the variance (power) of the noise: higher noise power gives less correlated sources.

In this model we use a perfect communication channel. Of course this is not a realistic view, but we are concentrating on the source coding aspect of the whole system. Errors due to channel imperfections can be solved by adding channel coding or by looking at joint source-channel coding [27] and is outside the scope of our work.

A high level design model of the transmitter/encoder side of the construction is shown in Fig. 16.

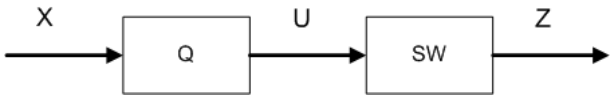


Figure 16: Transmitter side of system

A block diagram for the decoder is shown in Fig. 17

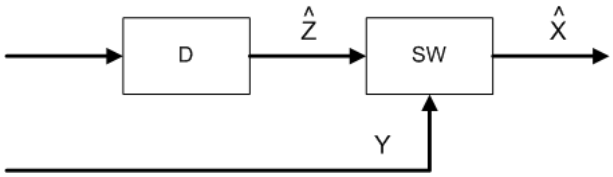


Figure 17: Receiver side of system

10.1 Hamming code implementation

First, a distributed source encoder with a $(7, 4)$ Hamming matrix is implemented. A parity-check matrix $\mathbf{H}$ is constructed consisting of 7 columns and $7 - 4 = 3$ rows. The

"generator" matrix used on the encoder side is then the transposed of $\mathbf{H}$, $\mathbf{G} = \mathbf{H}^T$. The parity-check matrix and its transposed are shown in Fig. 18. Note that the generator matrix as used here is not the same as a generator matrix in conventional LDPC coding used for error correction. The input ECG samples are quantized using an pdf optimized quantizer. Here the quantizer is optimized for a gaussian input distribution even though the ECG data are not gaussian distributed. But the gaussian approximation seems to be a good one in this case. The quantizer may be designed to any rate desirable. Here we quantize the input samples with a 7-bit quantizer to match it with the distributed encoder bitwise.

$$H = \begin{bmatrix} 1\,0\,0\,1\,1\,1\,0 \\ 0\,1\,0\,1\,1\,0\,1 \\ 0\,0\,1\,0\,1\,1\,1 \end{bmatrix} \qquad H^T = \begin{bmatrix} 1\,0\,0 \\ 0\,1\,0 \\ 0\,0\,1 \\ 1\,1\,0 \\ 1\,1\,1 \\ 1\,0\,1 \\ 0\,1\,1 \end{bmatrix}$$

Figure 18: The parity-check matrix $mathbfH$ and its transpose

The active (quantized) codewords are then multiplied with $\mathbf{G}$ and transmitted to the decoder. The outcome set of encoded bits represents the cosets of the channel code. Many codewords belong to the same coset as explained in Section 6. In the $(7, 4)$ Hamming code there are 2^7 possible codewords and 2^3 different cosets. Thus there are 2^4 codewords that output the same encoded bit sequence, i.e. belong to the same coset. The aim is to design a *good* channel code with large minimum distance between the codewords in the coset. The codewords of coset 000 is shown in Fig. 19.

0000000	0100110	1000101	1100011
0001011	0101101	1001110	1101000
0010111	0110001	1010010	1110100
0011100	0111010	1011001	1111111

Figure 19: The codewords of coset 000 in a $(7, 4)$ distributed Hamming code

The decoder of the Hamming example is a maximum likelihood decoder. When it receives a bit sequence from the channel, it finds the coset (with the predefined amount of bits) and the codewords it contains. Then, using the side information Y (which is correlated to the original active codeword X), the maximum likelihood decoder searches through the codewords in the received coset to find the closest one (in some given metric). The resulting estimation $\hat{X}$ of X is then the output of the decoder. The overall goal of the system is to minimize $\hat{X} - X$ given some constraints on encoding complexity and delay (Slepian-Wolf encoder), and inside a fidelity criterion (quantizer). This error is thus a good measure on our system.

10.2 LDPC code implementation

Going over to LDPC coding we create larger parity-check matrices with lower density. This is because encoding on larger block lengths gives better performance according to Shannon's source coding theorem [28], and larger block lengths yields larger matrices. The parity-check matrix $\mathbf{H}$ is constructed randomly. The quantized codewords are then gathered in desired block-lengths (e.g. 10^3) and multiplied with $\mathbf{H}$ in the same fashion as in the Hamming example. The encoded bits represent a block of coset bit sequences and are sent over the channel for decosing and reconstruction.

The decoder uses an adapted version of the sum-product algorithm as described in Section 9 where the decisions are not only based on the received bit sequence and the parity-check constraints in the Tanner graph, but in addition it takes into consideration the correlated bit sequence of the side information Y. Due to lack of time during the work with this thesis, this could not be implemented and tested. If continued work on this subject is needed, this may an assignment for future students.

11 Simulation & Results

The implementation in Section 10 was tested for the Hamming code example with a compression rate of $n : n - k$, which in this case means a rate-3/7 encoder. The result is shown in Fig. 20 where the correlation is along the x-axis by means of the conditional entropy $H(X|Y)$, and the probability of error is along the y-axis. The error probability is on a codeword basis.

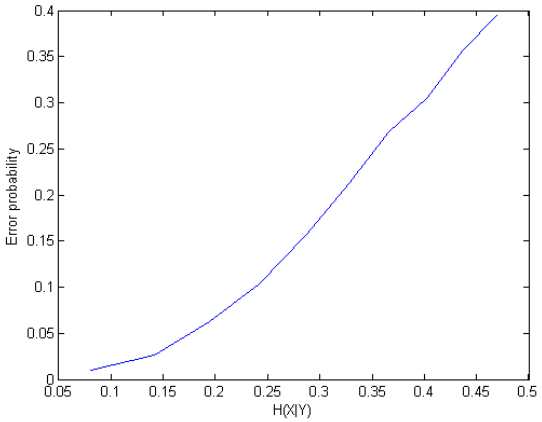


Figure 20: Simulation results

We see that this is not a very impressive result with error probability up to $4 \cdot 10^{-}1$, but then again it is not a very robust code construction. The ecoder works on blocks of only 7 bits, the channel code is not optimized for minimum distance coset construction and the decoder algorithm is not the sum-product iterative algorithm. The result would improve drastically if the adapted sum-product algorithm had been implemented.

Even though the error probability is high we see from Fig. 21 in the low correlation end of the test values, the reconstructed ECG signal (Fig. 21(a)) is not that unrecognizable from the original (Fig. 21(b)).

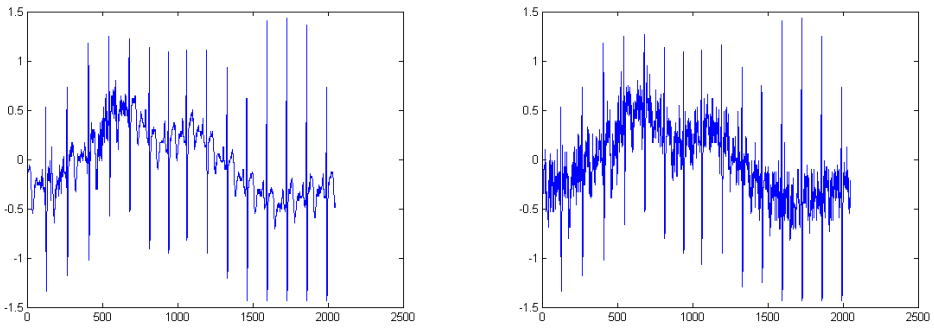
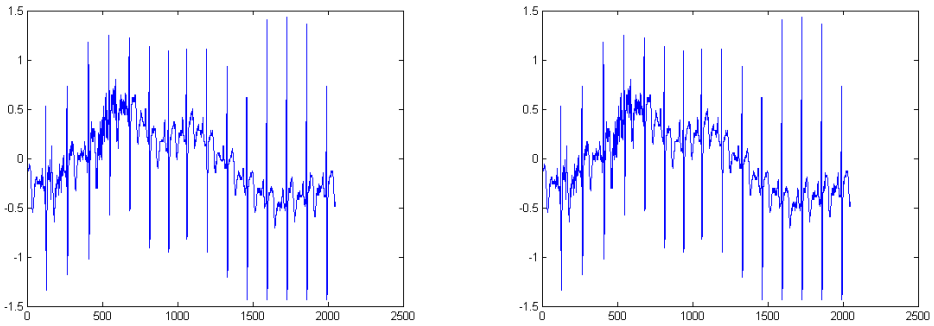


Figure 21: (a) Quantized ECG signal (b) Output with $H(X|Y) = 0,47$

Now in the other end of the results, where the correlation noise is lower, Fig. 22 shows that the output ECG signal is almost identical to the input signal even though the error percentage is quite high. One reason is that for low noise the estimated sample does not have the possibility to differ a lot from the real one, and with a high rate quantizer the distortion is not noticeable.



(a) (b)

Figure 22: (a) Quantized ECG signal (b) Output with $H(X|Y) = 0,08$

The LDPC code could not be simulated since the decoder was not implemented in time as mentioned before.

12 Conclusion

The main goal of this work was to use a clever coding framework to increase the lifetime of a sensor in a wireless sensor network. These sensors have to operate without the supply of any power, and the need to decrease the amount of data each sensor will have to transmit is crucial. Distributed source coding shows promising results in doing exactly this. We describe a way to do compression based on the correlation with other sources without the need for intersensor communication.

As a practical implementation we have used LDPC codes for compression. LDPC coding in its most known form of basic channel coding is still improving and has come as close as 0.0045dB to the Shannon limit [29]. When used in distributed source coding it has shown better performance than turbo coding and is approaching the Slepian-Wolf limit [12, 30]. In this thesis we only got to test the theory with a $(7, 4)$-Hamming code as the distributed encoder. This did not show a very good performance, but worked as a picture on how the techniques are deployed and utilized in the distrubuted source coding scheme.

Trying to illustrate the theory of this thesis in a more practical environment, we have used the distributed source code on a thought network of medical sensor nodes measuring ECG data on (in) a human body. This is just one piece in the whole puzzle of implementing compression based on correlation in a sensor network, but it may be an essential piece in lowering the energy consumption and hence increasing the lifetime of a sensor.

The work in this thesis may well be continued by other students interested in the area. Finalizing the decoder with an implementation of an enhanced sum-product algorithm will be the main goal. Expansions may be done in doing symmetric coding as described in Section 7 and including compression of not only two but multiple sources. This may then evolve to be a realistic model for distributed source coding in a sensor network

References

[1] V. Hsu, J. Kahn, and K. Pister, "Wireless communication for smart dust," 1998.

[2] L. Doherty, B. Warneke, B. Boser, and K. Pister, "Energy and performance considerations for smart dust," *Int. Journal of Parallel and Distributed Systems and Networks*, 2001.

[3] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *WSNA'02*, 2002.

[4] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, pp. 102–114, 2002.

[5] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inform. Theory*, pp. 471–480, 1973.

[6] A. Wyner, "On source coding with side information at the decoder," *IEEE Trans. Info. Theory*, vol. 21, pp. 294–300, 1975.

[7] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Trans. Info. Theory*, vol. 22, pp. 1–10, 1976.

[8] A. Wyner, "Recent results in shannon theory," *IEEE Trans. Inform. Theory*, pp. 2–10, 1974.

[9] S. Pradhan and K. Ramchandran, "Distributed source coding: Symmetric rates and applications to sensor networks," in *Proc. DCC'00*, 2000, pp. 363–372.

[10] ——, "Distributed source coding using syndromes (discus): Design and construction," *IEEE Trans. Inform. Theory*, pp. 626–643, 2003.

[11] S. Pradhan, J. Kusuma, and K. Ramchandran, "Distributed compression in a dense microsensor network," *IEEE Signal Proc. Magazine*, vol. 19, pp. 51–60, March 2002.

[12] D. Shonberg, K. Ramchandran, and S. Pradhan, "Distributed code constructions for the entire slepian-wolf rate region for arbitrarily correlated sources," in *Proc. DCC'04*, 2004.

[13] A. Aaron and B. Girod, "Compression with side information using turbo codes," in *DCC'02*, 2002.

[14] Y. Zhao and J. Garcia-Fras, "Data compression of correlated non-binary sources using punctured turbo codes," in *DCC'02*, 2002.

[15] J. Bajcsy and P. Mitran, "Coding for the slepian-wolf problem with turbo codes," in *ISIT'02*, 2002.

[16] J. Chou, S. Pradhan, and K. Ramchandran, "Turbo and trellis-based constructions for source coding with side information," in *DCC'03*, 2003.

[17] T. Cover and J. Thomas, *Elements of Information Theory.* Wiley, 1991.

[18] A. Gersho and R. Gray, *Vector Quantization and Signal Compression.* Kluwer, 1992.

[19] L. Lundheim and G. Øien, *Informasjonsteori, koding & kompresjon.* Tapir, 2005.

[20] J. Gibson, T. Berger, T. Lookabaugh, D. Lindbergh, and R. Baker, *Digital Compression for Multimedia*. Morgan Kaufmann, 1998.

[21] R. Zamir, S. Shamai, and U. Erez, "Nested linear/lattice codes for structured multiterminal binning," *IEEE Trans. Inform. Theory*, pp. 1250–1276, 2002.

[22] J. Barros and S. Servetto, "On the rate-distortion region for separate encoding of correlated sources," in *ISIT'03*, 2003.

[23] T. Berger, "Multiterminal source coding," in *The Information Theory Approach to Communications*, G. Longo, Ed. Springer-Verlag, 1977.

[24] B. Rimoldi and R. Urbanke, "Asynchronous slepian-wolf coding via source-splitting," in *Proc. IEEE Symp. Info. Theory*, 1997, p. 271.

[25] B. Sklar and F. Harris, "The abcs of linear block codes," *IEEE Signal Proc. Mag.*, pp. 14–35, 2004.

[26] R. Gallager, *Low Density Parity-Check Codes*. MIT Press, 1963.

[27] F. Hekland, "A review of joint source-channel coding," 2005.

[28] C. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423 and 623–656, 1948.

[29] S. Chung, G. Forney, and T. Richardson, "On the design of low-density parity-check codes within 0.0045 db of the shannon limit," *IEEE Commun. Letters*, pp. 58–60, 2001.

[30] A. Liveris, Z. Xiong, and C. Georghiades, "Compression of binary sources with side information at the decoder using ldpc codes," *IEEE Comm. Letters*, vol. 6, pp. 440–442, 2002.