

# Coding at the Sample Level for Data Hiding: Turbo and Concatenated Codes

Félix Balado and Fernando Pérez-González

*Departamento de Tecnologías das Comunicacions*, University of Vigo  
Lagoas-Marcosende, E-36200 Vigo, Spain

## ABSTRACT

The performance of data hiding techniques in still images may be greatly improved by means of coding. In previous approaches repetition coding was firstly used to obtain  $N$  identical Gaussian channels over which block and convolutional coding were used. But knowing that repetition coding can be improved we may turn our attention to coding forwardly at the sample level. Bounds for both hard and soft decision decoding performance are provided and the use of concatenated coding and turbo coding for this approach is explored.

**Keywords:** DCT, image watermarking, performance analysis, turbo codes, concatenated codes

## 1. INTRODUCTION

Data hiding schemes working at the sample level are those in which each codeword symbol is modulated into a single sample of the host image. All the data hiding schemes that we have considered so far<sup>1,2</sup> relied in replicating each information bit in different samples of the image to gain enough SNR so that a low probability of error could be achieved. This initial repetition, aided by an interleaving stage, allowed to obtain identical Gaussian channels for each bit being transmitted. After this step, standard digital communications analysis for additive Gaussian noise channels could be employed to predict the performance of that watermarking system. Also channel coding techniques were used for the improvement of this scheme; each codeword symbol was therefore implicitly being sent over several image samples.

The repetition technique closely resembles diversity techniques used in digital communications over fading channels<sup>3</sup>. Diversity can be seen as a simple form of block coding (with only two codewords) for which soft decision decoding is extremely simple. The good performance achieved by this almost trivial way of coding, also known as repetition coding, opens the door for more sophisticated types of codes that provide additional improvements over the basic scheme. For this purpose we will investigate coding at the sample level, which differs from the approach taken in most of the prevalent literature.

## 2. SAMPLE-LEVEL MODULATION SCHEME

In the following, variables in bold letters represent vectors whose elements will be referenced using the notation  $\mathbf{x} = (x[1], \dots, x[L])$ . An image  $\mathbf{x}$  can also be regarded to as an unidimensional sequence with  $L$  elements (samples) defined in the domain in which watermarking is performed. Note that the particular ordering of these samples is arbitrary and will have no effect on the final theoretical results.

The model we are following for the watermark generation only differs from former methods in the fact that now there is no initial diversity stage. The basic procedure of modulation is as follows. First, a sequence  $\mathbf{s}$  is produced by a pseudo-random generator initialized to a state depending on the key  $K$ , that is known only to the copyright owner and to the authorized recipient. This sequence of i.i.d. samples is constrained to have zero mean and unit variance at each sample, that is,  $E\{s[i]\} = 0$ ,  $E\{s^2[i]\} = 1$ ,  $\forall i = 1, \dots, L$ .

In order to guarantee invisibility, the sequence  $\mathbf{s}$  is multiplied sample by sample by a *perceptual mask*  $\alpha$  which results after analyzing the original image with a psychovisual model which takes into account the different sensitivity of the human visual system (HVS) to alterations in the different elements of  $\mathbf{x}$ . Then,  $\alpha^2[i]$  can be seen as the peak

---

Further author information:

Félix Balado: E-mail: fiz@tsc.uvigo.es

Fernando Pérez-González: E-mail: fperez@tsc.uvigo.es

energy constraint that the watermark must satisfy at the  $i$ -th sample for being imperceptible. The perceptual mask is computed differently depending on the domain chosen for watermarking and on the specific properties of the HVS that are being taken into account.

After  $\mathbf{s}$  and  $\boldsymbol{\alpha}$  are multiplied, the resulting sequence modulates the information symbols  $\mathbf{b}$  as we will detail next. Suppose that we want to hide a binary message  $\mathbf{m}$  of length  $N$ . After mapping it into a vector of antipodal information symbols  $\mathbf{b} = 2\mathbf{m} - \mathbf{1}$ , a very simple way of hiding it into  $\mathbf{x}$  would be to alter each sample in an amount with magnitude  $\alpha[i]$  and sign dependent on the hidden symbol, so that the perceptual constraint is met. In this way, the watermark would be obtained as  $w[i] = b[i]\alpha[i]s[i]$ ,  $i = 1, \dots, L$ , and  $N = L$  bits of information could be conveyed.

Unfortunately, such a simple scheme would result useless because generally  $\alpha[i] \ll x[i]$ , and consequently there would be a large probability of error for each hidden bit. So in general we must add redundancy to  $\mathbf{m}$  before modulation, using for it some channel coding method that transforms  $k$  bits of the message into a codeword  $\mathbf{q}$  of length  $n > k$ . Depending on the coding method and the available  $L$  samples we will have room for hiding  $\lfloor L/n \rfloor$  codewords, what fixes the maximum length for the conveyed information  $\mathbf{m}$  in  $N = k\lfloor L/n \rfloor$  bits; we will neglect possible border effects in our analysis. The actual codewords used for modulation  $\mathbf{q}^{(l)}$  are mapped into antipodal ones  $\mathbf{c}^{(l)} \in \mathcal{C} = \{\mathbf{c}_r\}_{r=1}^{2^k}$ ,  $l = 1, \dots, \lfloor L/n \rfloor$ , and concatenated as  $\mathbf{c}_T = (\mathbf{c}^{(1)} \mid \dots \mid \mathbf{c}^{(\lfloor L/n \rfloor)})$ .

In order to maximize the uncertainty associated to each bit the set of  $L$  available samples is scrambled using a key-dependent random permutation. For the sake of keeping the notation in previous papers, we will represent this interleaving by  $L$  one-sample-wide, non-overlapping subsets  $\{S_i\}_{i=1}^L$ . In this way, the watermark generation equation is

$$w[j] = c_T[i]\alpha[j]s[j], \quad j \in S_i. \quad (1)$$

Finally, the watermarked image that conceals the secret information is obtained as  $\mathbf{y} = \mathbf{x} + \mathbf{w}$ .

## 2.1. Considerations on the Domain

Optimal extraction (decoding) structures require the existence of statistical models for  $\mathbf{x}$ , because without them only *heuristic* approaches are possible. Same applies to performance analysis, that is not feasible without statistical knowledge of the image (channel).

Considering that no initial diversity stage is used in sample-level data hiding, it becomes obvious that it is not possible to resort to the central limit theorem approximation that permitted a Gaussian assumption for the channel. Besides, there is a lack of good statistical models for images in the spatial domain, and thus we must look for other domains where statistical knowledge is available.

Several authors have recently proposed to approximate the AC coefficients in the discrete cosine transform (DCT) by independent random variables following a generalized Gaussian probability density function (pdf), given by the expression

$$f_x(x[i]) = A[i] e^{-|\beta[i] x[i]|^\lambda}, \quad (2)$$

where both  $A[i]$  and  $\beta[i]$  can be expressed as a function of  $\lambda$  and the standard deviation  $\sigma[i]$

$$\beta[i] = \frac{1}{\sigma[i]} \left( \frac{\Gamma(3/\lambda)}{\Gamma(1/\lambda)} \right)^{1/2}, \quad A[i] = \frac{\beta[i] \lambda}{2 \Gamma(1/\lambda)}. \quad (3)$$

The standard deviations  $\sigma[i]$  are allowed to vary to permit a more flexible modeling of the coefficients at different frequencies. The use of this statistical characterization was independently proposed by Hernández et al.<sup>4</sup> and Barni et al.<sup>5</sup> to derive optimal extraction functions in the DCT domain.

The specific model we will use for  $\mathbf{x}$  is the Laplacian ( $\lambda = 1$ ) because, together with the Gaussian ( $\lambda = 2$ ), is the simplest in the family of generalized Gaussian models, producing at the same time relatively accurate results. It is possible to extend most of the results that will be presented here to other values of  $\lambda$  but sometimes no closed-form expressions exist and usage of numerical methods is required. The details on the adequacy of the Laplacian model and on the derivation of the perceptual mask  $\boldsymbol{\alpha}$  in the DCT domain can be found in previous works<sup>6</sup>.

### 3. DECODING STRATEGIES

In this section we will discuss the performance of hard and soft decoding for the proposed sample-level modulation scheme, assuming no attacks on  $y$  (e.g.: loss of synchronization). We will rely on the statistical characterization of  $x[i]$  and consider the key  $K$  as deterministic, and we will also assume that the sequences  $\alpha$  and  $\sigma$  can be characterized by means of a stochastic process with joint pdf  $f_{\alpha,\sigma}(\alpha, \sigma)$ . Moreover, by virtue of the key-dependent interleaving used in the watermark insertion stage, we can consider both sequences  $\alpha[i]$  and  $\sigma[i]$  as i.i.d. With these assumptions it is clear that all key-dependent partitions will produce the same results.

#### 3.1. Hard-Decision Decoding

In this case, an independent decision is taken for each codeword symbol. As the pdf of  $x[j]$  is symmetric then the symbol-by-symbol ML decision threshold will be set at the origin. Let us assume without loss of generality that the  $i$ -th codeword symbol takes the value  $+1$  and that  $s[j] = 1$ ,  $j \in \mathcal{S}_i$ . Then, the probability of error  $p(i | \alpha[j], \sigma[j])$  for this symbol is

$$p(i | \alpha[j], \sigma[j]) = P\{x[j] + \alpha[j] < 0\}, \quad j \in \mathcal{S}_i. \quad (4)$$

The probability  $p(i | \alpha[j], \sigma[j])$  can be easily evaluated for the case in which  $x[j]$  follows a Laplacian distribution, yielding

$$p(i | \alpha[j], \sigma[j]) = \frac{1}{2} e^{-\sqrt{2} \frac{\alpha[j]}{\sigma[j]}}. \quad (5)$$

When this probability is averaged over the sequence  $\alpha$  of perceptual masks and the sequence  $\sigma$  of variances, it becomes independent of the index of the transmitted symbol. Thus, the BSC cross-over probability  $p$  for this decoder can be written as

$$p = \int_{\alpha, \sigma} \frac{1}{2} e^{-\sqrt{2} \frac{\alpha}{\sigma}} f_{\alpha, \sigma}(\alpha, \sigma) d\alpha d\sigma. \quad (6)$$

An approximation to solve Eq. (6) consists in using an ergodic assumption on  $\alpha$  and  $\sigma$  that allows to approximate  $p$  as

$$p = \frac{1}{L} \sum_{j=1}^L \frac{1}{2} e^{-\sqrt{2} \frac{\alpha[j]}{\sigma[j]}}. \quad (7)$$

Once  $p$  is available, the Bhattacharyya bound on the bit error probability ( $P_b$ ) and an approximation<sup>7</sup> can be obtained using

$$P_b \leq \frac{2^{k-1}}{2^k - 1} \sum_{m=2}^{2^k} [4p(1-p)]^{w_m/2}, \quad P_b \simeq \frac{1}{n} \sum_{m=t+1}^n m \binom{n}{m} p^m (1-p)^{n-m}, \quad (8)$$

where  $w_m$  is the Hamming weight of the  $m$ -th codeword and  $t$  the code error-correcting capability. Note that since the value of  $p$  will be in general quite large, use of powerful codes will be essential in order to achieve good performance.

#### 3.2. Soft-Decision Decoding

The optimal ML detector for the Laplacian case<sup>6</sup> would decide codeword  $\hat{c} \in \mathcal{C}$  such that

$$\hat{c} = \arg \max_{c \in \mathcal{C}} \sum_{i=1}^n c[i] \cdot \left. \frac{|y[j] + \alpha[j]s[j]| - |y[j] - \alpha[j]s[j]|}{\sigma[j]} \right|_{j \in \mathcal{S}_i}. \quad (9)$$

Regarding the bit error probability, in order to use the union bound, the probability of error between two codewords has to be computed. Assuming that  $c_1$  and  $c_2$  are the only existing codewords and that  $c_1$  is actually sent, we should determine the probability of error conditioned on a certain sequence of perceptual masks  $\alpha$  and variances  $\sigma$ , that is,  $P_2(\alpha, \sigma) \triangleq P(c_1 \rightarrow c_2 | \alpha, \sigma)$  and where  $x$  is the only random variable in the system. Collecting all these considerations, we can state that there will be a block error (i.e., the decoder will decide  $c_2$ ) iff

$$\sum_{i=1}^n (c_2[i] - c_1[i]) \cdot \left. \frac{|y[j] + \alpha[j]s[j]| - |y[j] - \alpha[j]s[j]|}{\sigma[j]} \right|_{j \in \mathcal{S}_i} > 0. \quad (10)$$

In App. A a bound on the probability of error between two-codewords  $P(\mathbf{c}_1 \rightarrow \mathbf{c}_2)$  is derived. For linear codes, once the set of two-codewords error bound probabilities between  $\mathbf{c}_1$  and any other codeword are available, we can bound the global codeword error by means of

$$P_c \leq \sum_{r=2}^{2^k} P(\mathbf{c}_1 \rightarrow \mathbf{c}_r). \quad (11)$$

#### 4. CHANNEL CODING FOR LOW SIGNAL-TO-NOISE RATIOS

Sample-level coding for data-hiding applications faces the problem of the very low *per sample SNR* (see Sect. 5) that is usually encountered, which is a direct consequence of the imperceptibility constraint. Then, although there is a potential advantage in using better channel codes than simple repetition, it is also true that powerful codes are required in order to achieve a small probability of error. Unfortunately, for moderately large values of  $d_{\text{free}}$  ( $d_{\text{min}}$ ) in the convolutional (block) code, decoding has a tremendous complexity. This difficulty also arises in deep-space communications where transmitted power is as well severely limited too<sup>8</sup>, so several solutions to overcome it have been explored yet. We will see in this section how to adapt them to the particular problem of sample-level data hiding.

##### 4.1. Concatenated Codes

This approach was first proposed by Forney<sup>9</sup> and can be summarized by Fig. 1 for a typical data hiding application. Note that only one level of concatenation is shown, but the idea is easily generalized to any level. In our context, the inner code would be a binary block  $(n, k)$  or convolutional  $(k/n)$  code and the outer code would be a block code with an alphabet of  $q$  symbols (typically, a Reed-Solomon with  $q = 256$ ) so that the output of the latter is compatible with the input of the former. With concatenation it is possible to achieve a  $d_{\text{min}}$  which is the product of the minimum distances of the two concatenated codes; on the other hand, decoding complexity is merely that of each individual code.

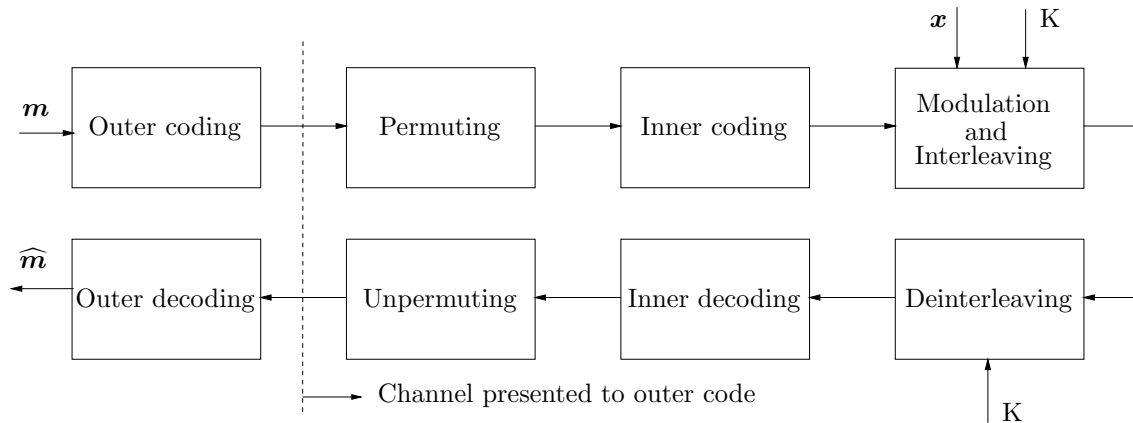


Figure 1. Concatenated coding and decoding scheme

An important element of many concatenated codes is the *permuter*<sup>7,10\*</sup>, which is a device that produces a systematic permutation of the symbols at the output of the outer encoder. The function of this permutation is to spread error bursts that may appear at the output of the inner decoder, making it easier for the outer decoder to correct them. These error bursts are common at the output of convolutional decoders. Note that permuter memory (size), a frequent concern when designing permuters for communications applications due to delay issues, is not so critical here due to the availability of the entire image.

\*We use the term ‘permuter’ to denote an interleaver used for coding purposes, in order to avoid mistakes with the key-dependent interleaver in the modulation stage.

As we see in Fig. 1 the concatenation just creates a less error-prone channel that is then presented to the outer code. So performance analysis involves iterating two calculations: first, the cross-over probability  $p$  of a decoding error (either with soft or hard decision) at the output of inner decoder is computed (or upper-bounded) as in Sect. 3. This probability characterizes the channel seen by the outer code, thus we may obtain the overall bit error probability  $P_b$  by substituting it into an expression like (8) with the outer code parameters. Of course, this approach to performance assumes a perfect permuter between the codes that makes the decisions of the inner decoder look to be independent.

Also, notice that the use of coding combined with diversity, as done in previous approaches and recalled in Sect. 1, can also be seen as a form of concatenation in which the inner code is simply a repetition code; interestingly, this choice allows for soft decoding of the outer code.

## 4.2. Turbo Codes

These codes are based in concatenation together with iterative decoding. In Fig. 2 we can see a standard turbo coder. It consists in the parallel concatenation of two binary rate recursive systematic convolutional (RSC) encoders separated by a  $k$ -bit *non-uniform* permuter; more permuters and encoders could be added in the same way. We will only consider here the case where these encoders are equal and of rate  $1/t$ , having one systematic output and  $t - 1$  parity outputs. Being  $\mathbf{m}$  the  $k$  bits of information fed to the encoder the output codeword will have the form

$$\mathbf{q} = (\mathbf{q}^s \mid \mathbf{q}^{p_{1,1}} \dots \mathbf{q}^{p_{1,t-1}} \mid \mathbf{q}^{p_{2,1}} \dots \mathbf{q}^{p_{2,t-1}}) = (\mathbf{q}^s \mid \mathbf{q}^{p_1} \mid \mathbf{q}^{p_2}), \quad (12)$$

where  $\mathbf{q}^{p_{r,t}}$  is the  $l$ -th parity output vector of constituent coder  $r$  and  $\mathbf{q}^s = \mathbf{q}^{s_1} = \mathbf{q}^{s_2} = \mathbf{m}$  the systematic output, that is not repeated because it is the same for both encoders. All of these subvectors of  $\mathbf{q}$  are of length  $k$ ; consequently, the codeword length will be  $n = k(2t - 1)$  and the turbo coder rate  $1/(2t - 1)$ . This rate can be raised using a puncturing mechanism, but in watermarking applications this is not in general desirable due to the low dynamic range available for encoding.

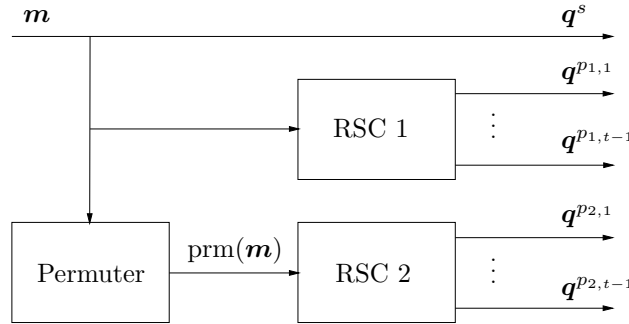


Figure 2. Turbo coder scheme

The calculated rate also sets an upper limit on the permuter size. Considering that the performance of turbo decoding is known to increase with permuter size (see Sect. 4.2.2) and given that it is not a critical parameter for the same reasons discussed in Sect. 4.1, we will choose the permuter to be as large as possible, i.e.  $k = \lfloor L/(2t - 1) \rfloor$ . As a result, only one codeword  $\mathbf{q}$  of length  $n$  will be modulated in its antipodal form  $\mathbf{c}$  into the image to yield  $\mathbf{y}$ , as explained in Sect. 2, and the number of hidden information bits will be  $N = k$ .

### 4.2.1. Iterative decoding

As permuter sizes can be very high (e.g.:  $k > 1000$ ) the complexity of a ML decoder would be exorbitant for the  $2^k$  possible sequences. In addition, it is difficult to straightforwardly use Viterbi algorithm, because the permuter complicates greatly the structure of a hypothetical turbo code trellis. So turbo codes are considered more like big  $(n, k)$  block codes, being its decoding accomplished by using iteratively each one of the two decoders. In each iteration maximum a posteriori (MAP) decoding is performed using *a priori* soft reliability information given by the previous decoding iteration. In this way, complexity is approximately only that of the constituent codes. We shall outline next the iterating procedure.

For the sake of simplifying the notation, we will assume that  $\mathbf{y}$ ,  $\boldsymbol{\alpha}$  and  $\mathbf{s}$  are deinterleaved from the key-dependent scrambling used in modulation, so that the samples in these three vectors match the ones with same indices in  $\mathbf{c}$ . Besides, we will decode the binary vector message  $\mathbf{m}$  in its antipodal version  $\mathbf{b}$ .

In the symbol-by-symbol MAP decoder the decision  $\hat{b}[i] = +1$  is taken if  $p(b[i] = +1 | \mathbf{y}) > p(b[i] = -1 | \mathbf{y})$ . In other words,  $\hat{b}[i] = \text{sign}\{L(b[i])\}$  where  $L(b[i])$  is the log *a posteriori* probability ratio:

$$L(b[i]) \triangleq \log \left( \frac{p(b[i] = +1 | \mathbf{y})}{p(b[i] = -1 | \mathbf{y})} \right). \quad (13)$$

Using the code's trellis it can be written as

$$L(b[i]) = \log \left( \frac{\sum_{W^+} p(w', w, \mathbf{y})}{\sum_{W^-} p(w', w, \mathbf{y})} \right), \quad (14)$$

where  $W^+$  ( $W^-$ ) is the set of encoder state transitions  $w' \rightarrow w$  caused by a symbol input  $b[i] = +1$  ( $-1$ ), being  $w'$  the state at sample  $i - 1$  and  $w$  the state at sample  $i$ . Assuming a memoryless transmission channel  $\mathbf{x}$ , the probabilities in Eq. (14) can be put as the product of three independent terms:

$$p(w', w, \mathbf{y}) = p(w', \mathbf{y}[j < i]) \cdot p(w, \mathbf{y}[i] | w') \cdot p(\mathbf{y}[j > i] | w) = \lambda_{w'}[i - 1] \cdot \gamma_{w' \rightarrow w}[i] \cdot \mu_w[i], \quad (15)$$

where the  $\mathbf{y}[i]$  subvector is defined as

$$\mathbf{y}[i] = (y^s[i], y^{p^{1,1}}[i], \dots, y^{p^{1,t-1}}[i], y^{p^{2,1}}[i], \dots, y^{p^{2,t-1}}[i]), \quad i = 1, \dots, k, \quad (16)$$

whose elements are taken from the appropriate samples of  $\mathbf{y}$ . The BCJR algorithm<sup>11</sup> shows how to recursively compute the  $\lambda$  and  $\mu$  probabilities as

$$\lambda_w[i] = \sum_{w'} \gamma_{w' \rightarrow w}[i] \lambda_{w'}[i - 1] \quad \mu_w[i - 1] = \sum_w \gamma_{w' \rightarrow w}[i] \mu_w[i], \quad (17)$$

initializing  $\lambda_0[1] = \mu_0[k] = 1$  and  $\lambda_w[1] = \mu_w[k] = 0$  for states  $w \neq 0$  (assuming the encoders start and end at zero state). As for the branch transition probabilities  $p(w, \mathbf{y}[i] | w')$ , they can be written as

$$\gamma_{w' \rightarrow w}[i] = p(\mathbf{y}[i] | b[i]) \cdot p(b[i]). \quad (18)$$

Using the log *a priori* probability ratio  $L^e(b[i]) \triangleq \log [p(b[i] = +1)/p(b[i] = -1)]$  the *a priori* probability  $p(b[i])$  can be expressed as

$$p(b[i]) = K_i \exp \left( b[i] L^e(b[i])/2 \right). \quad (19)$$

Taking into account that  $c^s[i] = b[i]$  the conditioned probability in (18) for decoder  $r$  can be written as

$$p(\mathbf{y}[i] | b[i]) = Q \exp \left( - \frac{|y^s[i] - b[i] \alpha^s[i] s^s[i]|}{\sigma^s[i]} \right) \exp \left( - \sum_{l=1}^{t-1} \frac{|y^{p^{r,l}}[i] - c^{p^{r,l}}[i] \alpha^{p^{r,l}}[i] s^{p^{r,l}}[i]|}{\sigma^{p^{r,l}}[i]} \right). \quad (20)$$

Notice that for this step it is necessary to know the statistical characterization of the channel (image) and that, for the same input symbol  $b[i]$ , the parities  $c^{p^{r,l}}[i]$  depend on the encoder state. Using  $\gamma_{w' \rightarrow w}$  we may substitute now (15) into the numerator and the denominator of (14). Noting that (19) and the first exponential in (20) can be taken out of the summations over  $W^+$  and  $W^-$ , and that  $Q$  and  $K_i$  cancel out at the numerator and denominator, we may write

$$L(b[i]) = \left( \frac{|y^s[i] + \alpha^s[i] s^s[i] - |y^s[i] - \alpha^s[i] s^s[i]|}{\sigma^s[i]} \right) + L^e(b[i]) + \log \left( \frac{\sum_{W^+} \lambda_{w'}[i - 1] \cdot \gamma_{w' \rightarrow w}^e[i] \cdot \mu_w[i]}{\sum_{W^-} \lambda_{w'}[i - 1] \cdot \gamma_{w' \rightarrow w}^e[i] \cdot \mu_w[i]} \right), \quad (21)$$

where  $\gamma_{w' \rightarrow w}^e[i]$  equals the second exponential in (20). The first summand in (21) is known as *channel value*, while the second is the *a priori* information received from the previous decoding iteration and the third is the reliability *extrinsic* information to be used as *a priori* data at the next decoding iteration. For the decoder  $r = 1$  we can write (21) as

$$L_1(b[i]) = L_c[i] + L_{21}^e(b[i]) + L_{12}^e(b[i]). \quad (22)$$

We can see the decoding procedure loop in Fig. 3; the first time  $L_{21}^e$  is initialized to zero. This feedback explains the term 'turbo' first given<sup>12</sup> to this decoding procedure.

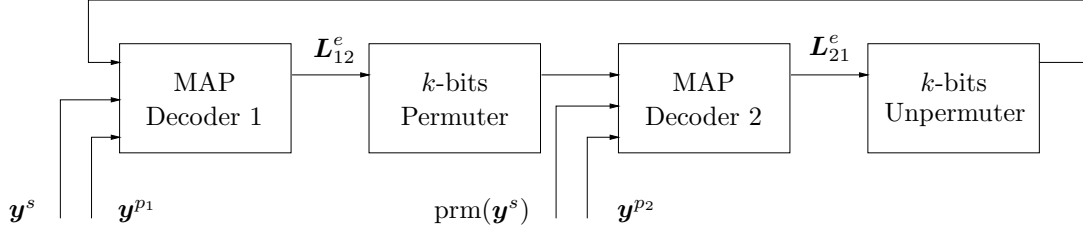


Figure 3. Turbo decoder scheme

#### 4.2.2. Performance

It has been demonstrated that turbo codes are near-optimal, in the sense that they can provide almost error-free decoding after some iterations for a SNR near the Shannon limit. In watermarking applications the main trouble we face is the lack of convergence of the decoding process, so we shall select a low enough rate and choose constituent coders<sup>13</sup> that yield optimal overall convergence properties. Note that, for a given rate, the RSC coders with highest  $d_{\text{free}}$  will not necessarily give the best performance.

It is important to stress that the permuter must rearrange the data symbols in a non-uniform (pseudo-random) fashion. In this way a high overall (turbo)  $d_{\text{min}}$  is achieved, enhancing the performance of the turbo decoder. For large values of  $k$  and low SNR's any pseudo-random permuter will work.

The  $P_b$  of the turbo decoder for the Laplacian channel can be approximately bounded following the same reasoning described by Ryan<sup>14</sup> and using the bound on the probability of two-codewords error with soft decoding (40) derived in App. A:

$$P_b \lesssim \max_{h \geq 2} \left\{ \frac{h n^{(h)}}{k} P(\mathbf{c}_1 \rightarrow \mathbf{c}_2) \Big|_{d_{1,2}=d_{\text{min}}^{(h)}} \right\}, \quad (23)$$

where  $n^{(h)}$  is the number of weight- $h$  messages  $\mathbf{m}$  that produce minimum weight codewords  $\mathbf{q}$ , and  $d_{\text{min}}^{(h)}$  is that minimum over the set of weight- $h$  input messages. Both values are function of the actual permuter used and have to be calculated by simulation. The maximum will usually be found for the first  $h \geq 2$  with  $n_h \neq 0$ . We observe in Eq. (23) the *permuter gain*: the  $P_b$  can be improved by using a bigger permuter.

## 5. EXPERIMENTAL RESULTS

We present in this section preliminary results obtained from applying concatenated and turbo coding to real data hiding cases. It is important that the performance results can be compared to those obtained with other methods like the former diversity ones. In order for the results to be independent from the image used we have proposed<sup>15</sup> to normalize the performance results using the *per sample SNR* (PSSNR), which is defined as the total available SNR divided by the available samples  $L$ . In this way the  $P_b$  vs.  $\text{PSSNR} \cdot L/N$  curves are similar to the  $P_b$  vs.  $E_b/N_0$  ones used in digital communications.

In the diversity approach the  $P_b$  curves were obtained varying the amount of diversity samples used and so varying the PSSNR and  $N$ . This strategy is no longer possible for we are using only one sample for hiding each information bit; as a result we will resort to scaling the perceptual mask  $\alpha$  for obtaining the  $P_b$  curves. Moreover, the PSSNR has to be redefined because, instead of having identical channel noise power and mean signal energy, we have now varying channel noise and energy for each codeword symbol. A reasonable (but not unique) option that we have used is

$$\text{PSSNR} = \frac{\sum_{i=1}^L \alpha^2[i]}{\sum_{i=1}^L \sigma^2[i]}. \quad (24)$$

In order to illustrate the performance of coding at the sample level we have carried out simulations using the well-known image *Lena* ( $256 \times 256$  pixels) and averaging the results over 20 different keys. Only the mid-range DCT frequencies were used for hiding information<sup>6</sup>, what results in  $L = 22528$  samples.

For the concatenated coding tests we used inner convolutional codes with rates  $(1/t)$  and hard decoding, and Reed-Solomon outer codes with  $q = 256$ ; these coders were separated by a block permuter that rearranged its input so that any burst of less than  $k_{\text{outer}}$  consecutive symbol errors resulted in errors isolated by at least  $n_{\text{outer}}$  symbols. In Fig. 4 (a) the results for some combinations of codes are shown. We found that a powerful inner code (high  $d_{\text{free}}$ ) was necessary for a proper functioning. Also it appears that Reed-Solomon codes with moderate  $n$  sizes perform better. It should be noted that when powerful codes are used there is a considerable increase in the Viterbi decoder complexity.

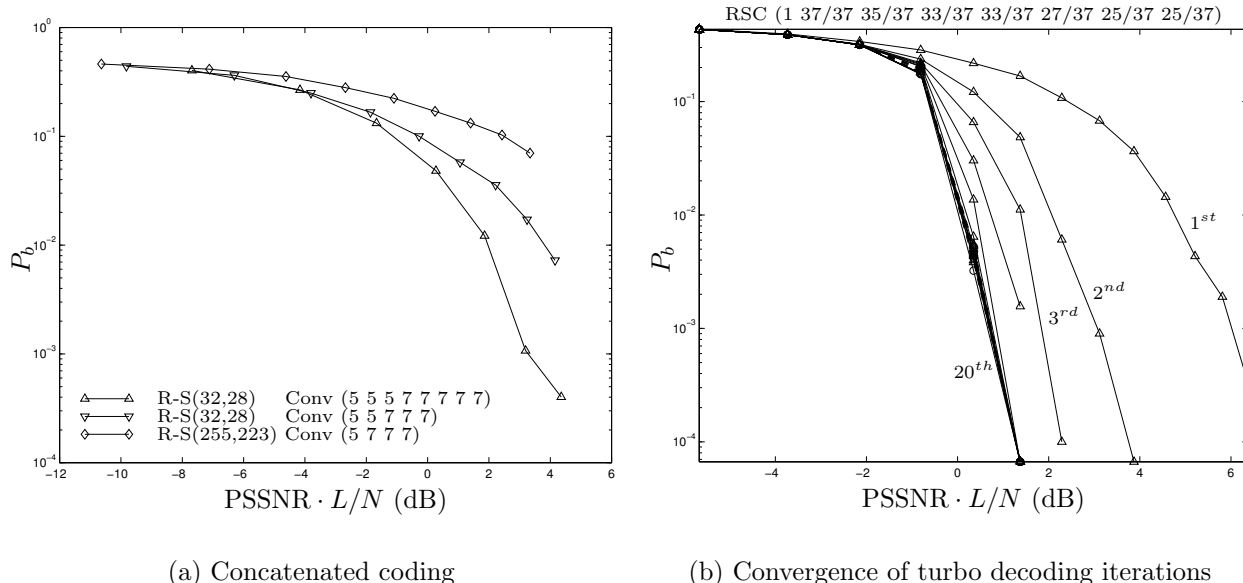


Figure 4.

As for the turbo decoding tests we may see in Fig. 4 (b) an example on how the  $P_b$  diminishes in each iteration for a given RSC. Ideally, RSC's with optimal properties should be used<sup>16,13</sup>; nevertheless we found that the optimum RSC's for turbo coding explored to the present had rates that were too high for our problem, so common convolutional codes with rates as low as  $1/8$  were implemented as RSC's. We can see that approximately at the 5<sup>th</sup> iteration the decoding has converged to the final values, getting for a  $\text{PSSNR} \cdot L/N$  just slightly higher than 1 dB a very good bit error probability of  $P_b = 10^{-4}$ . Notice also that for this case the hidden information payload is as high as 1500 bits in only 22528 host samples; besides, the visibility threshold lies around 4 dB in the  $\text{PSSNR} \cdot L/N$  curves so the hidden watermark is not perceptible. Interestingly, not only decoding performance is much better than concatenated coding (as expected) but also decoding speed is higher, accomplishing 20 iterations in about 10 seconds for our implementation.

## 6. CONCLUSION AND FURTHER WORK

In this paper we have presented a watermarking scheme based in sample-level data hiding in the DCT-domain, making use of advanced coding techniques. This scheme outdoes those based in an initial diversity stage because we know that better codes than diversity are available. Making use of the statistical characterization of the host signal, theoretical bounds on the performance of the optimum decoder are derived, allowing us to predict on beforehand the suitability of a given image for data hiding.

Further investigations are required to firmly establish the PSSNR so as to determine the exact improvement over diversity methods. In addition, it would be very interesting to extend these methods to domains showing invariances to attacks such as geometrical transformations and warpings. The gain obtained with sample-level coding is a good starting point to compensate for capacity losses due to employing such domains (e.g.: Fourier-Mellin). More theoretical work in the determination of the capacity and on detection tests for sample-level data hiding is required likewise.

Another future possibility would be to use concatenation of turbo codes with BCH or even Reed Solomon codes<sup>17</sup> when a very low  $P_b$  is necessary. This is so because turbo codes use to present an error floor once a critical  $\text{PSSNR} \cdot L/N$  (lower for more powerful RSC's) is exceeded. An alternative to this concatenation is a careful permuter design.

## APPENDIX A. BOUND ON CODEWORD ERROR FOR SOFT DECISION DECODING

In this appendix we derive a bound for the two-codeword error probability in the DCT domain (Laplacian case), with coding at the sample level and soft-decoding. Without loss of generality we will assume that  $s[i] = \alpha[i]/|\alpha[i]|$ ,  $c_1[i] = -1$ ,  $\forall i = 1, \dots, n$ . Then  $y[i] = -\alpha[i] + x[i]$  will follow a Laplacian distribution of variance  $\sigma^2[i]$  centered at  $-\alpha[i]$  and, applying the fundamental theorem of statistics, we can see that the transformation  $z[i] = (|y[i] + \alpha[i]s[i]| - |y[i] - \alpha[i]s[i]|) / \sigma[i]$  will yield the fdp

$$f_{z[i]}(z) = f_{z[i]}^\Delta(z) * \delta(z + r[i]/2), \quad (25)$$

$$f_{z[i]}^\Delta(z) = \frac{1}{2} \delta(z) + \frac{1}{2\sqrt{2}} e^{\frac{-z}{\sqrt{2}}} [u(z) - u(z - r[i])] + R[i] \delta(z - r[i]), \quad (26)$$

where

$$r[i] = 4 \frac{|\alpha[i]|}{\sigma[i]}, \quad R[i] = \int_{r[i]}^{+\infty} \frac{1}{2\sqrt{2}} e^{\frac{-z}{\sqrt{2}}} dz = \frac{1}{2} e^{\frac{-r[i]}{\sqrt{2}}}. \quad (27)$$

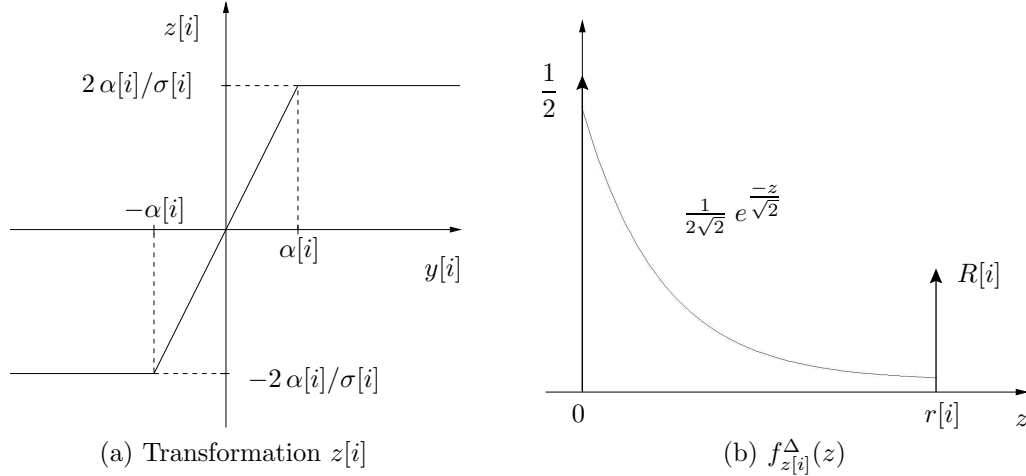


Figure 5.

$\delta(\cdot)$ ,  $u(\cdot)$  and  $*$  stand respectively for the Dirac's delta function, the unit step function and the convolution integral. The transformation and the subsequent fdp are depicted in Fig. 5.

Therefore the probability that inequality (10) holds when  $c_1$  is correct can be written as

$$P_2(\boldsymbol{\alpha}, \boldsymbol{\sigma}) = P \left\{ \sum_{j \in \mathcal{J}} z[j] > 0 \right\}, \quad (28)$$

where the set  $\mathcal{J} = \{j \in \mathcal{S}_i / c_1[j] \neq c_2[j]\} = \{j_1, \dots, j_{|\mathcal{J}|}\}$ , and its cardinality  $|\mathcal{J}|$  is just the Hamming distance  $d_{1,2}$  between  $c_1$  and  $c_2$ . Recalling the independence hypothesis, the fdp of the summation will be obtained as the convolution of all individual probability density functions, so (28) becomes

$$P_2(\boldsymbol{\alpha}, \boldsymbol{\sigma}) = \int_0^{+\infty} f_{z[j_1]}(z) * \dots * f_{z[j_{|\mathcal{J}|}]}(z) dz = \int_{r_T}^{+\infty} f_{z[j_1]}^\Delta(z) * \dots * f_{z[j_{|\mathcal{J}|}]}^\Delta(z) dz, \quad (29)$$

where

$$r_T = \sum_{j \in \mathcal{J}} \frac{r[j]}{2}. \quad (30)$$

As a closed-form expression of the multiple convolution inside the integral is not possible, we will upper bound  $P_2(\boldsymbol{\alpha}, \boldsymbol{\sigma})$  by considering the fdp

$$h(z) = \frac{1}{2} \delta(z) + \frac{1}{2\sqrt{2}} e^{\frac{-z}{\sqrt{2}}} u(z). \quad (31)$$

Given an arbitrary fdp  $g(z)$ ,  $h(z)$  exhibits the following property

$$\int_{\gamma}^{+\infty} f_{z[i]}^{\Delta}(z) * g(z) dz \leq \int_{\gamma}^{+\infty} h(z) * g(z) dz, \quad \forall i, -\infty \leq \gamma \leq +\infty. \quad (32)$$

We can prove this as follows

$$\begin{aligned} p_L &= \int_{\gamma}^{+\infty} f_{z[i]}^{\Delta}(z) * g(z) dz = \int_{\gamma}^{+\infty} \int_{-\infty}^{+\infty} f_{z[i]}^{\Delta}(x) g(z-x) dx dz = \\ &= \int_{\gamma}^{+\infty} \left\{ \int_{-\infty}^{r[i]^-} f_{z[i]}^{\Delta}(x) g(z-x) dx + \int_{r[i]^-}^{+\infty} f_{z[i]}^{\Delta}(x) g(z-x) dx \right\} dz = \\ C + \int_{\gamma}^{+\infty} \int_{r[i]^-}^{+\infty} f_{z[i]}^{\Delta}(x) g(z-x) dx dz &= C + \int_{\gamma}^{+\infty} \int_{r[i]^-}^{+\infty} R[i] \delta(x-r[i]) g(z-x) dx dz = \\ &= C + R[i] \int_{\gamma}^{+\infty} g(z-r[i]) dz = C + R[i] \int_{\gamma-r[i]}^{+\infty} g(z) dz. \end{aligned} \quad (33)$$

For the right hand side of inequality (32), taking into account that  $h(z) = f_{z[i]}^{\Delta}(z)$  for  $z < r[i]$ , we have

$$\begin{aligned} p_R &= C + \int_{\gamma}^{+\infty} \int_{r[i]^-}^{+\infty} h(x) g(z-x) dx dz = C + \int_{r[i]^-}^{+\infty} h(x) \int_{\gamma}^{+\infty} g(z-x) dz dx = \\ C + \int_{r[i]^-}^{+\infty} h(x) \int_{\gamma-x}^{+\infty} g(z) dz dx &\geq C + \int_{r[i]^-}^{+\infty} h(x) \int_{\gamma-r[i]}^{+\infty} g(z) dz dx = C + R[i] \int_{\gamma-r[i]}^{+\infty} g(z) dz = p_L. \quad \square \end{aligned} \quad (34)$$

The inequality in (34) stems from the fact that a fdp is a non-negative function. Therefore, using (32) iteratively in Eq. (29) it follows that

$$P_2(\boldsymbol{\alpha}, \boldsymbol{\sigma}) \leq \int_{r_T}^{+\infty} \underbrace{h(z) * \dots * h(z)}_{d_{1,2} - 1 \text{ convolutions}} dz = \int_{r_T}^{+\infty} q(z) dz. \quad (35)$$

Now we may proceed to evaluate the convolutions and the subsequent integral. Taking the unilateral Laplace transform of  $h(z)$  we get

$$H(s) = \frac{1}{2} + \frac{(2\sqrt{2})^{-1}}{s + \frac{1}{\sqrt{2}}}. \quad (36)$$

So the transform of  $d_{1,2} - 1$  convolutions of  $h(z)$  with itself will be

$$Q(s) = H(s)^{d_{1,2}} = \sum_{i=0}^{d_{1,2}} \binom{d_{1,2}}{i} \left(\frac{1}{2}\right)^{d_{1,2}-i} \frac{(2\sqrt{2})^{-i}}{\left(s + \frac{1}{\sqrt{2}}\right)^i} = \sum_{i=0}^{d_{1,2}} \frac{D_i}{\left(s + \frac{1}{\sqrt{2}}\right)^i}, \quad (37)$$

whose inverse transform is

$$q(z) = D_0 \delta(z) + \sum_{i=1}^{d_{1,2}} D_i \frac{z^{i-1}}{(i-1)!} e^{\frac{-z}{\sqrt{2}}} u(z). \quad (38)$$

Then

$$P_2(\boldsymbol{\alpha}, \boldsymbol{\sigma}) \leq \sum_{i=1}^{d_{1,2}} \frac{D_i}{(i-1)!} \int_{r_T}^{+\infty} z^{i-1} e^{-\frac{z}{\sqrt{2}}} dz = \sum_{i=1}^{d_{1,2}} \binom{d_{1,2}}{i} \frac{2^{-d_{1,2}}}{(i-1)!} \Gamma\left(i, \frac{r_T}{\sqrt{2}}\right) = \sum_{i=1}^{d_{1,2}} \binom{d_{1,2}}{i} \frac{2^{-d_{1,2}}}{(i-1)!} \Gamma\left(i, \sqrt{2} \sum_{j \in \mathcal{J}} \frac{|\alpha[j]|}{\sigma[j]}\right), \quad (39)$$

where  $\Gamma$  is the incomplete gamma function<sup>†</sup>. For obtaining the bound on  $P(\mathbf{c}_1 \rightarrow \mathbf{c}_2)$  we have to integrate (39) over  $\boldsymbol{\alpha}$  and  $\boldsymbol{\sigma}$ , recalling the hypothesis of i.i.d. for all  $\alpha[j]$  and  $\sigma[j]$ :

$$P(\mathbf{c}_1 \rightarrow \mathbf{c}_2) = \int_{\boldsymbol{\alpha}, \boldsymbol{\sigma}} P_2(\boldsymbol{\alpha}, \boldsymbol{\sigma}) f_{\boldsymbol{\alpha}, \boldsymbol{\sigma}}(\boldsymbol{\alpha}, \boldsymbol{\sigma}) d\boldsymbol{\alpha} d\boldsymbol{\sigma}. \quad (40)$$

The exact calculation is a bit tedious but can be accomplished by integrating iteratively the first summand in the recursive definition of  $\Gamma$ , that can be solved as follows:

$$\int_{\boldsymbol{\alpha}, \boldsymbol{\sigma}} -\left(\sqrt{2} \sum_{j \in \mathcal{J}} \frac{|\alpha[j]|}{\sigma[j]}\right)^m \exp\left\{-\sqrt{2} \sum_{j \in \mathcal{J}} \frac{|\alpha[j]|}{\sigma[j]}\right\} f_{\boldsymbol{\alpha}, \boldsymbol{\sigma}}(\boldsymbol{\alpha}, \boldsymbol{\sigma}) d\boldsymbol{\alpha} d\boldsymbol{\sigma} = -\sum_{k_1=0}^m \sum_{k_2=0}^{k_1} \cdots \sum_{k_{d-1}=0}^{k_{d-2}} \binom{m}{k_1} \binom{k_1}{k_2} \cdots \binom{k_{d-2}}{k_{d-1}} I_{m-k_1} I_{k_1-k_2} \cdots I_{k_{d-2}-k_{d-1}} I_{k_{d-1}}, \quad (41)$$

with  $d = d_{1,2}$  and

$$I_l = \int_{\alpha, \sigma} \left(\sqrt{2} \frac{|\alpha|}{\sigma}\right)^l e^{-\sqrt{2} \frac{|\alpha|}{\sigma}} f_{\alpha, \sigma}(\alpha, \sigma) d\alpha d\sigma, \quad (42)$$

which can be approximated as in (7) (notice that only  $d_{1,2}$  different integrals/summations like this have to be calculated).

Assuming ergodicity on  $\alpha$  and  $\sigma$  an easier way to approximate the bound on  $P(\mathbf{c}_1 \rightarrow \mathbf{c}_2)$  in practice would be to average Eq. (39) over a certain number of randomly chosen sets  $\{\mathcal{J}_k\}$ , each one containing  $d_{1,2}$  different indices.

## ACKNOWLEDGMENTS

This work has been partly supported by the *Xunta de Galicia* under project PGIDT99 PX132203B and by the European project Certimark (Certification of Watermarking Technologies), IST-1999-10987.

## REFERENCES

1. J. R. Hernández, F. Pérez-González, J. M. Rodríguez, and G. Nieto, "Performance analysis of a 2d-multipulse amplitude modulation scheme for data hiding and watermarking of still images," *IEEE J. Select. Areas Commun.* **16**, pp. 510–524, May 1998.
2. J. R. Hernández and F. Pérez-González, "Statistical analysis of watermarking schemes for copyright protection of images," *Proceedings of the IEEE* **87**, pp. 1142–1166, July 1999.
3. J. G. Proakis, *Digital Communications, 2nd Ed.*, Mc Graw-Hill, New York, 1989.
4. J. R. Hernández, F. Pérez-González, and M. Amado, "Improving DCT-domain watermarking extraction using generalized gaussian models," in *Proc. of the COST #254 workshop on Intelligent Communications and Multimedia Terminals*, pp. 23–26, (Ljubljana, Slovenia), November 1998.
5. M. Barni, F. Bartolini, V. Capellini, A. Piva, and F. Rigacci, "A MAP identification criterion for DCT-based watermarking," in *Proc. European Signal Processing Conference*, vol. I, pp. 17–20, (Rhodes, Greece), September 1998.

---

<sup>†</sup> $\Gamma(n, x) = \int_x^{+\infty} z^{n-1} e^{-z} dz = -x^{n-1} e^{-x} + (n-1) \Gamma(n-1, x)$

6. J. R. Hernández, M. Amado, and F. Pérez-González, "DCT-domain watermarking techniques for still images: Detector performance analysis and a new structure," *IEEE Trans. on Image Processing* **9**, pp. 55–68, January 2000.
7. B. Sklar, *Digital Communications. Fundamentals and Applications*, Prentice-Hall International Editions, 1988.
8. J. Yuen, M. Simon, W. Miller, C. Ryan, D. Divsalar, and J. Morakis, "Modulation and coding for satellite and space communications," *Proceedings of the IEEE* **78**, pp. 1250–1266, July 1990.
9. G. D. Forney, *Concatenated Codes*, MIT Press, Cambridge, MA, 1966.
10. S. G. Wilson, *Digital Modulation and Coding*, Prentice-Hall, Upper Saddle River, New Jersey, 1996.
11. L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory* **IT-20**, pp. 284–287, March 1974.
12. C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes (1)," in *Proc. IEEE Int. Conf. on Communications*, pp. 1064–1070, (Geneva, Switzerland), May 1993.
13. S. Benedetto, R. Garello, and G. Montorsi, "A search for good convolutional codes to be used in the construction of turbo codes," *IEEE Trans. Communications* **46**, pp. 1101–1105, September 1998.
14. W. E. Ryan, "A turbo code tutorial," tech. rep., New Mexico State University, 1998.
15. F. Pérez-González, J. R. Hernández, and F. Balado, "Approaching the capacity limit in image watermarking: A perspective on coding techniques for data hiding applications." Accepted for final publication in *Signal Processing*, Elsevier, 2000.
16. J. D. Andersen, "Selection of code and interleaver for turbo coding," in *Proc. First ESA Workshop on Tracking, Telemetry and Command Systems*, ESTEC, (The Netherlands), June 1998.
17. J. D. Andersen, "Turbo coding for deep space applications," in *Proc. 1995 IEEE Int. Symp. on Information Theory*, (Whistler, Canada), September 1995.