LDPC Coded Watermarks for Audio:

Algorithms Tolerant to Compression and Desynchronization

Raúl Martínez-Noriega

電気通信大学

電気通信学研究科

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Engineering

March 2011

LDPC Coded Watermarks for Audio:

Algorithms Tolerant to Compression and Desynchronization

Approved by supervisory committee:

Chairperson:	Prof. Yamaguchi Kazuhiko	(山口　和彦　先生)
Member:	Prof. Ohta Kazuo	(太田　和夫　先生)
Member:	Prof. Kawabata Tsutomu	(川端　勉　先生)
Member:	Prof. Sakiyama Kazuo	(崎山　一男　先生)
Member:	Prof. Koda Hiromu	(小田　弘　先生)

概要

LDPC符号を用いた音楽用電子透かし：圧縮と同期破りに耐性のあるアルゴリズム

マルチネズ　ノリエガ　ラウル

電気通信大学

電子透かしは、ディジタル著作物の内部に情報を秘匿するプロセスである。例えばオーディオ・画像・動画に対して、ある程度の頑強さをもって埋め込まれる情報と著作物は一体の分離できないコンテンツとなる。電子透かしは、広告への利用からセキュリティ目的まで、その応用は広がっている。その中でもっとも一般的なものはディジタル著作物の不正利用を防止である。本論文では、オーディオデータに電子透かしに入れることと、自動的に同期可能な電子透かしの復元を扱う。オーディオデータへの電子透かし埋め込みを対象とし、高い埋め込みデータ量、MPEG-1 Audioレイヤー III（以下、MP3と呼ぶ）64kbpsという強い圧縮にも頑強な保持される電子透かしに焦点を当てた。また埋め込まれた電子透かしの同期による抽出は未解決の問題として検討を加えた。本研究で提案する同期の問題の解決方法は電子透かしのみならず通信全般への解決手法も提案するものである。電子透かしは、透かしが埋め込まれる、言い換えれば伝送されるという、ある種の通信であり、雑音を付加されたコンテンツから復元（つまり通信における受信）されていると見ることができる。しかし、これまでの研究ではそうした通信と透かし埋め込みの対比に着目した研究はほとんど行われていなかった。この視点によれば高信頼の埋め込み（伝送）を実現する電子透かしは、そのほかの通信システムと同様に誤り制御符号の適用により達成しうるのは明らかである。しかしながらこれまでの電子透かしの研究ではほとんど、この事実を利用していない。頑強な電子透かしを作り出す新しい方向はこの論文によって示される。計算が複雑でなく、大きなペイロードをもつが、埋め込み手法としては弱い電子透かしであるQuantization Index Modulation (QIM) を埋め込みアルゴリズムとし、近年強力な誤り訂正符号として注目されているLow-Density Parity-Check (LDPC)を組み合わせる方法に取り組んだ。電子透かし方式の解析評価においては、透かしを頑強に埋め込むことができる可聴周波数を検討した。また基本的なQIM法から派生したQIM型の埋め込み手法に対してガウス雑音やMP3圧縮に対する特性を評価した。さらに繰り返し符号

とLDPC符号による連接符号化によるLDPC符号の復号性能向上について検討している。研究結果として、64kbpsの品質のMP3圧縮に対して頑丈に埋め込むことができる電子透かし方式を実現している。またそのペイロードは（229.7ビット／秒）であり、これまで発表されたどのロバストなオーディオ電子透かしより高い埋め込み性能を有している。この電子透かしのペイロードの大きさは、広告のようなアプリケーションを考えることができるし、電子指紋方式の様な長いデータを埋め込む必要がある場合にも有効である。提案する電子透かしの品質を主観評価実験により行ったところ、最高5点中、平均で4点を超える良い品質を実現している。（ここで5点は原音とまったく区別がつかないことを意味している。）通信における同期は通常同期パターンをヘッダとしていれることで行われることが多い。しかし、このアプローチは最適とは言い難い。我々は、この同期用ヘッダを必要としないLDPC符号に基づく新しい自己同期アルゴリズムを提案する。このアルゴリズムは単に電子透かしに有効なだけでなく、一般的な通信システムに対する有効性も評価している。LDPC符号の符号長が増えると、提案する同期復号アルゴリズムは同期用ヘッダを用いる場合の復号性能に近づいた。例えば、符号長504の符号化率1/2LDPC符号で、我々のアルゴリズムと同期ヘッダを用いる方式はEb/N0=4 dBの点で同じ復号誤り率性能を実現する。またこのアルゴリズムに関する理論的な諸性能を定式化し記述している。また、復号特性を定める理論的限界を、符号長の小さいLDPC符号に対する最尤推定法復号を用いて示した。以上により、頑強な電子透かし方式と自己同期のアルゴリズムがオーディオに対する電子透かしというアプリケーションにおいて統合され、これまでにない、頑強さと多量のデータ埋め込める新しい電子透かし方式が実現できた。

Abstract

LDPC Coded Watermarks for Audio: Algorithms Tolerant to Compression and Desynchronization

by

Raúl Martínez Noriega

The University of Electro-Communications

Professor Kazuhiko Yamaguchi, Chair

Digital watermarking is the process of hiding information inside digital work, e.g. audio, image or video, with certain robustness in such a way that the information and the digital work are inseparable. Applications ranging from advertising until security-related can be benefited with digital watermarking, however the most popular and still potential application of digital watermarking is preventing illegal uses of digital work.

Audio watermarking and synchronization are the concerns of this dissertation. The interest on audio watermarking is focused in high embedding rates robust to strong compression like MPEG-1 Audio layer III (MP3) at 64 kbps. Synchronization is an open problem to digital watermarking, however our study on synchronization offers solutions not only to digital watermarking but general transmissions.

Watermarking is a sort of communication where the watermark is embedded/transmitted, contaminated with noise and retrieved/received from the digital work. Therefore reliable embedding/transmission on watermarking, as any other communication scheme, can be achieved with error-control codes (ECC). However, almost no previous study exploits this fact.

A new direction about the generation of robust watermarks is introduced in this dissertation. The algorithm combines low-complex, weak and high-payload watermarking algorithms like quantization index modulation (QIM) with powerful ECC like low-density parity-check (LDPC) codes.

The analysis on watermarking comprises audio frequencies where the watermark can be robustly embedded, behavior of different QIM-based watermark algorithms under Gaussian noise and compression, the repercussions of concatenating repetition codes with LDPC codes

and smart use of the channel to improve the decoding of the LDPC code. Thus, we introduce proposals which embed a coded-watermark in frequency domain. We explain how the statistics of noise, used by the LDPC decoder and generated due to MP3 compression, can be computed with accuracy improving the reliability of the decoded bits.

The result is the highest embedding payload, 229.7 bits per second, to an audio watermark algorithm with robustness to compression with quality of 64 kbps. This result allows audio watermarking to be used in applications like advertising or information carrier. The watermarked audio also achieves good audio quality, obtaining a score higher than 4 out of 5, where 5 represents indistinguishability with the original audio file, in both subjective and objective tests.

Synchronization is highly important in scenarios, including watermarking, where the information is transmitted as blocks of bits. Usually synchronization can be achieved using pilot bits, however this approach is by far non-optimal. We introduce a new self-synchronous decoding algorithm based on the cyclically permutable characteristics of LDPC codes. The algorithm does not need pilot symbols and its performance approaches synchronized transmissions when the code length is increased. For example, with a half-rate LDPC code with length of 504 the performance of our algorithm and synchronized transmission is the same for $E_b/N_0 = 4$ dB.

Theoretical results which include the scope, restrictions and the error-performance bound are also presented. The impact of this algorithm goes further than watermarking and therefore the algorithm is analyzed from the point of view of general communications.

Finally, watermarking and the self-synchronous algorithm were combined in a practical application of audio watermarking producing a new algorithm which has higher embedding capacity and more robustness than the original algorithm proposed by Lie in 2005.

Contents

概要 VII
Abstract IX
List of Figures XV
List of Tables XVII

1 Introduction **1**
1.1 Retrospective 2
1.1.1 Research progress 2
1.1.2 Commercial applications 6
1.2 Dissertation overview 7

2 Foundations **11**
2.1 Digital watermarking 11
2.1.1 Applications of audio watermarking 13
2.2 Watermarking is not cryptography 14
2.3 Quantization index modulation 14
2.3.1 Dither modulation 15
2.3.2 Spread-Transform dither modulation 16
2.3.3 Distortion-Compensation 17
2.4 Digital audio 18
2.4.1 Audio watermarking singularity 19
2.5 Attacks 20
2.5.1 MPEG-1 layer III (MP3) 20
2.5.2 Clipping attack 21
2.6 Low-density parity-check codes 22
2.6.1 Construction 22
2.6.2 Encoding 23
2.6.3 Decoding: Sum-product algorithm 23

3 Audio watermarking **27**
3.1 Embedding on frequency domain 28
3.1.1 Discrete wavelet transform 28
3.1.2 Framework of the embedding 29
3.1.3 Framework of the detection 29
3.2 QIM-based watermarking 30

3.2.1 Dither modulation . . . 31
3.2.2 Spread-Transform dither modulation . . . 33
3.3 Increasing the watermark robustness with ECC . . . 35
3.3.1 Dither modulation with LDPC codes . . . 36
3.3.2 Spread-Transform dither modulation with LDPC codes . . . 38
3.4 General analysis of QIM-based algorithms . . . 40
3.5 Further improvements . . . 44
3.6 Conclusion . . . 49

4 High payload watermarks for MP3 compression 51
4.1 Repercussions of MP3 on wavelet domain . . . 53
4.2 High payload audio watermark algorithm . . . 54
4.2.1 Embedding . . . 54
4.2.2 Decoding . . . 55
4.3 Embedding capacity tolerant to MP3 . . . 58
4.3.1 Semi-blind decoding . . . 58
4.3.2 Blind decoding . . . 60
4.4 Increasing the embedding capacity with erasures. . . . 62
4.5 Comparison with related algorithms . . . 65
4.6 Audio quality tests . . . 66
4.7 Conclusion . . . 67

5 Self-synchronous decoding with LDPC codes 69
5.1 Codes for synchronization . . . 71
5.1.1 Comma-free codes . . . 71
5.1.2 Cyclically permutable codes . . . 72
5.2 Problem definition and statements . . . 72
5.3 Preliminaries . . . 73
5.4 Self-synchronous decoding algorithm . . . 74
5.4.1 Fundamentals of synchronization capability . . . 77
5.4.2 Complexity . . . 80
5.4.3 Restrictions . . . 80
5.5 Performance over BPSK transmissions with Gaussian noise . . . 81
5.6 Toward the bound: Maximum likelihood decoding . . . 83
5.7 Codes and its CP-distance . . . 85
5.7.1 Codes with large CP-distance . . . 85
5.7.2 Codes with small CP-distance . . . 86
5.8 Conclusion . . . 87

6 Self-synchronizable audio watermarking based on LDPC coding 89
6.1 Lie's watermark algorithm . . . 90
6.1.1 Encoding the watermark . . . 92
6.1.2 Synchronization . . . 92

6.2 Self-synchronizable audio watermarking 93
6.2.1 Watermark embedding 93
6.2.2 Watermark detection 94
6.3 Results and comparisons 94
6.4 Conclusion 96

7 Concluding remarks **97**
7.1 Reflections about future research 98

List of contributions **101**

References **105**

Acknowledgements **111**

Author biography **112**

List of Figures

2.1 Generic block diagram of a watermark system. 12
2.2 Variation of Gaussian-Shannon channel. 12
2.3 Watermarking as communications with side information. 13
2.4 Concerns of cryptography and watermarking. 14
2.5 Design of secure systems using layers. 15
2.6 Characteristics of waves. 18
2.7 PCM converts analog audio signals into digital ones. 19
2.8 ISO/MPEG-1 layer III (MP3) encoder. 21
2.9 Clipping attack removes random samples from the cover work. 22
2.10 Tanner graph of the parity-check matrix H of the Hamming code $(7, 4)$. . . . 23

3.1 Decomposition of the audio file using discrete wavelet transform. 29
3.2 General embedding scheme on frequency domain for audio files. 30
3.3 General detection scheme on frequency domain for audio files. 30
3.4 Using repetition codes before embedding 31
3.5 DM robustness using repetition codes with different length l. 32
3.6 Performance of DM and DCDM over AWGN channel. 33
3.7 Comparison between DM and ST audio watermarking. 34
3.8 Performance of ST and STDC over AWGN channel. 35
3.9 General scheme using LDPC codes in watermarking. 36
3.10 Comparison of DM with LDPC codes and uncoded DM. 38
3.11 DCDM^c performs better using $\alpha < \alpha^*_{opt}$. 39
3.12 Performance of ST using LDPC codes and uncoded ST. 40
3.13 Performance of coded DCST. 40
3.14 Comparison of the best results using the LDPC code E. 42
3.15 Comparison of our proposals using LDPC code E and BCH codes. 43
3.16 Robustness against MP3 compression and low-pass filtering. 43
3.17 General block diagram of adaptive dither modulation. 45
3.18 Discrete wavelet packet transform of an audio signal until the 5^{th} level. 45
3.19 Probability density function of the soft-information extracted with DM. . . . 47
3.20 Performance of traditional DM and ADM for AWGN. 48
3.21 Histogram of the step size δ used in ADM. 48
3.22 Coded ADM with different kernels. 49

4.1 General diagram of the studied system. 51

4.2 Error percentage of the wavelet sub-bands after MP3 at 128 kbps. 54
4.3 General watermarking scheme. 55
4.4 Histograms of the soft-information after MP3 compression. 57
4.5 *Particular noise variance* with different noise variance. 59
4.6 Performance of the proposed method using semi-blind decoding. 60
4.7 Multiplexing pilot bits with the coded-watermark. 61
4.8 *Particular noise variance* using different number of pilot bits. 62
4.9 *Particular noise variance*: blind versus semi-blind decoding. 63
4.10 Outperform of semi-blind *particular noise variance*: MP3 compression. 64
4.11 Outperform of semi-blind *particular noise variance*: Low-pass filtering. 64
4.12 Best threshold t for different embedding rates. 64

5.1 Decoding with self-synchronization capability. 75
5.2 Hamming distance $d_H(\tilde{C}, \hat{C})$ after one iteration of SPA. 76
5.3 Decoding approach following an inverted *tree* technique. 78
5.4 Synchronized transmissions vs our self-synchronous algorithm: tiny LDPC code. 82
5.5 Recovering synchronization with a small half-rate LDPC code. 83
5.6 Synchronization with non CP codes: High-rate LDPC codes. 84
5.7 Recovering synchronization with deletions among different codewords. 85
5.8 Maximum likelihood decoding for our algorithm and its potential bounded code. 86
5.9 Short codes: Performance of equivalent codes in desynchronized transmissions. 88

6.1 GOS is composed of three different sections sec_1^i, sec_2^i and sec_3^i. 90
6.2 Structure of the embedding of Lie's method compared with ours. 93

List of Tables

3.1 Characteristics of the LDPC codes. 38
3.2 Summary of the proposals analyzed in Chapter 3 for audio watermarking. . . 41
3.3 Results of StirMark Audio benchmark. 44

4.1 Correlations related with the amount of noise in wavelet sub-bands 25-28. . . 65
4.2 BER and payload comparison of related algorithms against MP3 compression. 66
4.3 Subjective evaluation of audio quality based on *ITU-R BS.1116 standard*. . . 67
4.4 Objective evaluation of audio quality PEAQ. 67

5.1 Number of SPA iterations needed to decode a desynchronized codeword. . . . 80

6.1 BER results against MP3 compression and low-pass filtering. 95

1

Introduction

1994 was the beginning of the third-party services for processing on-line credit sales, the *e-commerce* was born. Internet is a huge market with the most rich variety of stuff including music. Selling audio or music through Internet brings many advantages like buying only a specific song, cheaper prices, and buying without leaving home. An example of this huge market is the well-known on-line store from a computer-related company who had sold more than 10 billion of songs by February 2010 [1].

This increasing growth of e-commerce has fostered research on techniques for preventing illegal uses of digital work such as audio, image, and video by means of cryptography and *watermarking*.

Watermarking describes methods to embed *auxiliary information* transparently into a work. The oldest register about watermarking dates back to 1282 in Italy, where paper watermarks were made by adding thin wire patterns to the paper molds. However the meaning and purpose of the earliest watermarks are uncertain.

Nowadays, watermarking is often used to increase the security of the bills. For example, held up a Japanese ¥1000 to the light. The echoed portrait that you are looking at the middle of the bill is called *watermark* and indicates the authenticity of the bill. The watermark is embedded directly into the paper during the papermaking process, and is therefore very difficult to forge. The purpose of the watermark is to carry information about the object in which it is hidden [2].

Digital watermarking was inspired by its analog counterpart. In this case the watermark, usually a stream of bits, is embedded into the digital work in such a way that the digital work and the watermark are inseparable unless the quality of the digital work loses its commercial value.

Applications of digital watermarking are many and cover a wide number of fields. For example, authenticity of images, extension of narrow bands in telephony speech, and fin-

gerprinting just to mention some. Furthermore, digital watermarking is not only applied to audio, image and video but to text and computer graphics as well.

Our interest in this dissertation is to develop watermark algorithms for audio tolerant to compression and desynchronization.

The first contribution proposes an audio watermark algorithm which achieves the highest embedding payload, 229.7 bps, robust to compression with quality of 64 kbps (similar to radio quality). The characteristics, payload and robustness, of this algorithm are suitable to applications like the insertion of advertisements in free samples of music or the embedding of subtitles or lyrics inside the audio file itself.

The second contribution is an algorithm to recover the synchronization in transmissions of block of bits. The algorithm was thought as a solution to clipping attack in audio watermarking. However, the impact of the proposal goes further than watermarking and it can be applied to any transmission based on iterative decoding of forward error correction techniques.

1.1 Retrospective

Although this dissertation is mainly related with audio watermarking, we impress a strong emphasis on low-density parity-check (LDPC) codes which are a class of forward error correction techniques. This philosophy is because we think that efficient watermark algorithms, in terms of payload and robustness, can be developed with highly noise-tolerant codes like LDPC codes and high-payload embedding functions. Moreover, LDPC codes are an important part in our proposal for synchronization. This proposal is useful for, but not restricted to, watermarking.

The literature review is related to watermarking, but we include important facts from error correcting coding[1] (ECC).

1.1.1 Research progress

Definitely error correcting coding is older than formal watermarking[2] technology. Nevertheless, very early applications of watermarking were developed around the same time when the first error correcting code, Hamming code, was invented.

Watermarking technology has at most 25 years old, and audio watermarking at least eight years less. However, in the last seven years, we have seen prominent audio techniques with good robustness-payload trade-off that show new insights toward more efficient algorithms.

LDPC codes were proposed long time ago but were forgotten until 1996. Recently, these

[1] Error control codes is an application of *coding theory* that enable reliable delivery of digital data over unreliable communication channels.

[2] We will use the term *watermarking* to refer specifically to digital watermarking.

codes have been selected, over *turbo codes*, to be in standards like the satellite transmission of digital television and WiMAX.

ECC and primitive watermarking: Before the 80s

The history of ECC started with the introduction of the Hamming codes in 1947, at or about the same time as the seminal work of Shannon in 1948 [3]. At this stage, watermarking was not yet invented. However the first commercial applications with the fundamental bases of watermarking were just about to be shown. The first record of watermark technology can be traced back to the 50s.

Several codes were discovered right after Shannon published the theoretical limits of reliable communications over noisy channels. Some of them are: Hamming codes (1947), Golay codes (1949), Reed-Muller codes (1954), convolutional codes (1955), BCH codes (1959) and Reed-Solomon codes (1960).

LDPC codes were proposed in 1962, [4], but were ignored because their decoding was complex to the computational capacity of that time. Nowadays, LDPC codes are decoding using a *soft-decoding* approach. However the bases of soft-decoding were developed in the 60s.

Heuristic watermarking: The 80s

The first attempts on watermarking research were in the 80s. Watermarking was not considered an independent technology and therefore all the research was basically heuristic approaches. The cover work and the watermark were considered to be independent signals and the constraint of fidelity-robustness was rawly adapted with a global power constraint at the embedder.

In 1981, Tanner [5] designed a method to represent codes by means of bipartite graphs. Later, these graphs will be important in the conception of efficient decoding algorithms for LDPC codes.

Watermarking as technology: The 90s

The middle of 90s was very important for watermarking as well as for LDPC codes. Watermarking was finally considered an independent technology and LDPC codes were rediscovered.

The understanding of watermarking at this point was more rigorous and with theoretical foundations. The published papers were increased not only in number but also in quality. Indeed, the fundamental bases of watermarking rely in several of those papers.

Theoretical studies modeled watermarking as a communication channel, where the cover work and further distortions were represented with a noisy channel in which the watermark

was transmitted through. Advances on fidelity included the aid of perceptual models, like the human visual system or the human auditory system, to adjust the watermark strength according to the cover work [6].

Parallel to the introduction of the perceptual modeling, important watermark algorithms were proposed: spread spectrum (SS) [7], [8], and patchwork [9]. At this point watermarking stop being exclusive to digital images, the proposals were extended to audio and video as well. However these first proposals only adapted algorithms from images to audio or video.

1996 is when finally the first proposal dedicated entirely to audio files was published by Boney [10].

Error correcting coding techniques had an explosive growing but none had achieved performances close to the theoretical limits until 1993, when *turbo coding* [11] was discovered. Three years later in 1996 MacKay and Neal show to the world that LDPC codes have also near-ideal performance using a new decoding algorithm based in the back-propagation principle and called *sum-product*, [12] and [13].

By the end of 90s, the watermarking model was revised and results from [14], [15] and [16] began to recognize that watermarking is more accurately modeled as *communication with side information* [17]. These new algorithms were referred as *informed watermarking* [18].

Finally, more rigorous quantitative measures of performance were introduced based on traditional false alarm and bit error rate techniques.

Watermarking and error control codes: Early 2000

The watermarking community was excited about the results of informed watermarking obtained in the last years. It was not long until 2001 that informed watermarking was refined with the introduction in [19], [20] and [21] of Costa's paper, *Writing on Dirty Paper* [22], to the watermarking community. This result is important to watermarking because the idea shows that the watermark performance is not affected by the cover work always that a proper encoding is conveyed on the watermark.

Quantization index modulation (QIM), a Costa-based algorithm, was introduced in 2001 by Chen [21]. QIM has been popular because is low-complex, has amenability to theoretical analysis and can achieve high payloads. Due to the impact of QIM, many variations have been proposed, being rational dither modulation [23] the most popular one. However no one has such performance benefit that justified the increasing of complexity.

This watermarking era is also the beginning of error correcting coding techniques as means of reliable retrieval of the watermark. The early studies were focused on BCH and convolutional codes, [24] and [25]. Only one year later in 2001, deeply studies about strategies of image watermarking using concatenations of BCH and repetition codes were published in [26]. In the same year, more advanced decoding techniques relying on soft-decision were

introduced by Pérez-González in [27]. However convolutional codes were the main topic, turbo coding was mentioned but not deeply analyzed. These techniques were mainly restricted to still images and there is not such deeply study dedicated to audio files.

An extensive analysis in 2002 about BCH and convolutional codes applied to watermarking [28] showed that these codes do not substantially improve the watermark performance due to watermarking channels are very noisy. This statement was reaffirmed three years later by Gu [29] who said that "for common signal processing including compression and noise corruption in images, BCH cannot improve the robustness of watermarking".

Spread spectrum was adapted to audio watermarking in 2003 by Kirovski [30] and in the same year, SS is combined with turbo codes [31] also for audio. This algorithm is robust to compression of 64 kbps and has payload of 21.6 bps.

Watermarking meets LDPC codes until 2004 when Bastug [32] used the codes to increase the embedding payload of SS watermarking in still images by twice more than the achievable with BCH codes and 2.7 times more than using repetition codes. The encoding/decoding of the LDPC code is the traditional one, the only novelty is the addition of the LDPC to watermarking.

LDPC codes are suitable for very noisy channels, like watermarking ones, because they have a near-optimal decoding algorithm with linear complexity and its minimum distance increases according to the code length. Moreover, LDPC codes have shown to approach the capacity of various channels and they have better performance than turbo codes for large codes.

Turbo codes and LDPC codes assume acknowledge of the statistics of the channel noise. This assumption is difficult to satisfy in watermarking because the watermark is exposed to a wide range of attacks which are unknown. However, Balado [33] proposed a solution applied to turbo codes in 2004. The drawback is that the analysis assumed only Gaussian noise which is not true to all the attacks in watermarking.

Characterization of compression and filtering for audio watermarking were studied by Cvejic *et ál.* also in 2004, [34]. Their algorithm obtains information from the attack and then decides the best place to embed the watermark, however the characterization is not exploited by any error control coding technique.

In addition, more accurate noise models were developed, particularly for the quantization noise which is very common after compression of images or audio files. The effect of quantization was rigorously modeled in [35] and the result showed that *dither modulation* was analogous to watermarking.

Robust and high-payload watermarking: 2005 ∼

These last years have brought many advances in audio watermarking as well as theoretical studies related with the joint iterative decoding of watermarking and error control codes. For example, attacks like Gaussian noise, scaling and desynchronization were characterized in [36] using the expectation-maximization algorithm for still images.

In 2006 Coumou [37] proposed the first algorithm of audio watermarking and LDPC codes. The proposal is aimed to synchronization rather than dealing with conventional attacks. However the synchronization is achieved with a synchronization code and the LDPC code do not present any novelty. In the same year, Dikici [38] investigated the design of the quantizers in QIM and its relation with the decoding of LDPC codes. This proposal is mainly theoretical, and real attacks are not considered.

A proposal on rational dither modulation and ECC was proposed in 2007, [39], however the authors again concluded than repetition codes are better fitted to audio watermarking than other codes. This conclusion is because they analyzed common ECC techniques like BCH, Golay or convolutional codes.

The vogue in audio watermarking is to develop robust algorithms with considerable payload. The most prominent results have been obtained by Deshpande [40] in 2008 and Bhat [41] in 2010. The former achieves 220 bps of payload, however it is only robust to compression of 96 kbps. The latter is a recent proposal which is robust to compression of 64 kbps and obtains 196 bps.

Image watermarking has shown mature algorithms that are already implemented in commercial applications. The research on audio watermarking is a bit behind compared with images but with a promising future. Trends on watermarking include watermarking protocols, reliable communications with forward error correction techniques, high payload techniques, among others.

1.1.2 Commercial applications

Commercial applications involving watermarks are not new and indeed, one of them proposed by Emil Hembrooke, was used for almost 30 years. In 1954, Hembrooke [2] of Muzak Corporation filed a patent for watermarking of musical works. The watermark was used to identify music using the Morse code. The embedding of the watermark was achieved with a notch filter centered at 1 kHz. The absence of energy at this frequency indicated that the notch filter had been applied and the duration of the absence used to code either a dot or a dash.

Profitable fields where watermarking has been widely used are advertisement monitoring and audience measurement. Both “Nielsen Media Research” and “Competitive Media Re-

porting" employ watermarking to verify how many times a certain advertisement is aired. "Verance Corporation" has introduced a service to monitor television and radio broadcast media using their audio watermarking technology.

Watermarks are also useful to control electronic devices. The advantage is that the control signal is embedded in the main signal, saving the bandwidth dedicated to the control one. Probably the first attempt of signaling embedding was patent by Tomberlin [42], who proposed to embed an auxiliary signal in the radio broadcasting which allows the receivers to remove the advertisements. Later, in 1962 Noller [43] of "Lynch Carrier Systems Inc." patented a system to control telephony equipment by adding a watermark to the speech signal.

Video watermarking has also its own applications, being interactive television the first goal. In 1976 Baer [44] of the Sanders Associates Inc. was issued a patent related with interactive television by means of the embedding of an auxiliary signal. Almost a decade later, in 1989, Interactive Systems Inc. was awarded a patent [45] for "commands to interactive television". An early application of this technology was in the synchronization of children's toys with live-broadcast or recorded video.

Dolby Labs [46] has also developed applications which involve watermarking. In 1981, they proposed "A sub-audible in-band tone system ... for identifying an FM stereophonic radio broadcast which is specially encoded, as with dynamic range improvement encoding or quadraphonic encoding, ... [and] which can control a visual display and switch in appropriate signal decoding circuitry when the tone is detected."

Nowadays Digimarc Inc. and its partners Media Science International (MSI), Civolution, Verance and MarkAny offer commercial watermarking solutions to audio, still image and video. MSI provides forensic audio watermarks and copy protection for CDs. Civolution offers tracking and monetization of audio content on the Internet. Verance provides solutions for copy protection within the DVD-audio, SD-audio and SDMI formats.

Moreover Digimarc Inc. expects to release an audio watermarking application in the first quarter of 2011.

1.2 Dissertation overview

This dissertation is about high-payload algorithms for audio watermarking robust to compression and desynchronization. In order to achieve our goals, watermarking is analyzed as a communication scheme in which the reliability of the delivered bits relies highly in forward error correction techniques.

The first contribution is an audio watermark algorithm with the highest payload tolerant to the lowest compression with commercial value (64 kbps) and the second contribution is

a solution to recover the synchronization with LDPC codes. The synchronization algorithm was firstly thought to endure clipping attacks in audio files, however we propose a generalized algorithm which can be applied not only to watermarking but general communications.

We start this dissertation in Chapter 2 with a briefly introduction to general topics related with our concerns. We explain the role of watermarking in secure systems, then we introduce in detail the main embedding algorithm, called QIM. Selected topics on digital audio, MPEG-1 layer III (MP3) compression and desynchronization due to clipping attack are also included. The Chapter ends with a description of LDPC codes.

Chapter 3 is dedicated to an extensive analysis of audio watermarking using QIM-based algorithms. We define a generic watermark scheme in frequency domain to analyze the properties of dither modulation (DM) and spread transform dither modulation (ST), both them with and without distortion-compensation (DC) post-processing. Further improvements are analyzed with the addition of LDPC codes. The robustness of the algorithms is tested against common signal processing like MP3 compression and low-pass filtering. Finally, a benchmark for audio watermarking named Audio StirMark was used to show the performance of the proposed methods.

Compressed audio is probably the audio format most played over the world, hence Chapter 4 concerns about audio watermark algorithms robust to MP3 with a compression rate of 64 kbps (equivalent to radio quality). Efficient frequencies in terms of robustness and imperceptibility were found after an analysis of the sub-bands produced by a wavelet packet decomposition of the audio file. Then, a new watermark algorithm is proposed by using dither modulation and coded-watermarks. The algorithm takes advantage of the analysis in Chapter 3 about LDPC codes. The embedding capacity of the algorithm was increased with a novel idea using erasures at the LDPC decoder resulting in the algorithm with highest embedding capacity for audio watermarking resistant to MP3 at 64 kbps. This Chapter also includes a comparison of performance between our algorithm and recently related proposals. Finally, the Chapter is closed with results about the sound quality of the watermarked audio.

Chapter 5 was motivated because the desynchronization produced by the clipping attack in audio watermarking. However, the Chapter is treated from the point of view of general communications because the proposal can be applied not only to watermarking but to general transmissions. We propose a novel algorithm which recovers the synchrony using LDPC codes after a continuous deletion of bits in the channel. The theoretical bases of encoding/decoding, scope, and restrictions are also explained. The good performance of the algorithm is shown through simulations over BPSK[3] channel with Gaussian noise and its errors are bounded with an idea based on cyclic codes. Reflections about the characteristics of the codes and its impact in the performance of our algorithm are addressed at the end of the Chapter.

[3]Binary phase-shift keying

The unification of audio watermarking and the self-synchronous algorithm, presented in the previous Chapter, is the topic of Chapter 6. We develop an improvement to Lie's algorithm for audio watermarking in time-domain. At the beginning of the Chapter, Lie's algorithm is introduced and we also explain its two disadvantages. Then, an application of the algorithm of Chapter 5 is proposed as improvement to Lie's algorithm. The advantages of this new algorithm are shown through simulations of both algorithms, Lie's and ours.

Finally, in Chapter 7 the ideas presented in this dissertation are concluded and we also include some suggestion to future research.

2

Foundations

Understanding of basic concepts is fundamental to conceive new ideas.

In this Chapter we describe general concepts which are needed to further comprehension of this dissertation. Among these topics are a formal definition of watermarking, similarities and differences between watermarking and cryptography, detailed explanation of the main embedding function, some facts about digital audio, compression, clipping attack, and an introduction to low-density parity-check (LDPC) codes.

2.1 Digital watermarking

Digital watermarking is formally defined as:

> *The process in which a signal, called watermark, is embedded in another signal, called cover work, with a certain robustness* [14].

In watermarking, unlike other data hiding techniques, the watermark always carries information about the cover work. Usually, the watermark is a binary stream and the cover work is audio, still images, or video.

A generic watermarking scheme is composed of two main parts: the embedding and the detection. The former combines the cover work with the watermark producing a unique entity, called watermarked work. The job of the detector is to retrieve the watermark as accurate as possible. If the watermarked work has not suffered distortion, then the extraction of the watermark is an easy job. However the watermarked work might suffer alterations which make difficult recovering the watermark.

In the 90s, watermarking was considered a sort of communications, see Fig. 2.1, in which the watermark (information to be transmitted) is embedded, the watermarked work is attacked (transmitted through noisy channels), and finally the watermark is retrieved (received). This analogy implies that reliable transmission in watermarking can be achieved

with the aid of coding theory, that is, using channel characterization and forward error correction techniques.

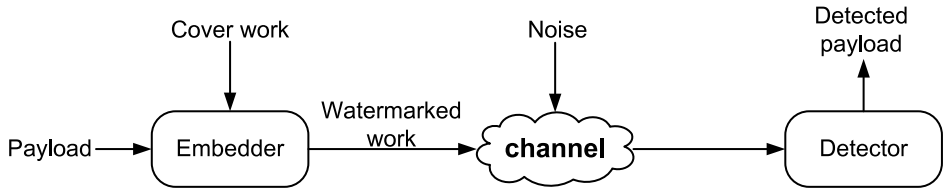


Fig. 2.1: Generic block diagram of a watermark system.

Later on, around the beginning of 2000, the watermarking model was refined and the conclusion was that watermarking is better fitted with a sort of channel firstly analyzed by Costa and called *communications with side information*

In 1983, Costa [22] studied a variation of the Gaussian-Shannon channel, depicted in Fig. 2.2, which has two sources of noise. One of them known by the encoder but them both unknown to the decoder.

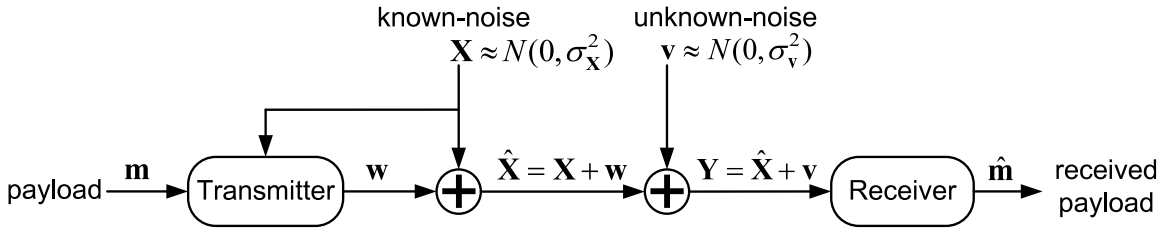


Fig. 2.2: Variation of Gaussian-Shannon channel.

Costa found that the channel capacity is:

$$C = \frac{1}{2}\log_2\Big(1 + \frac{\sigma_{\mathbf{w}}^2}{\sigma_{\mathbf{v}}^2}\Big), \tag{2.1}$$

where $\sigma_{\mathbf{w}}^2$ and $\sigma_{\mathbf{v}}^2$ are the variances of the source and the unknown-noise respectively. According to (2.1), the channel capacity is not affected by the known-noise and therefore, the channel of Fig. 2.2 is equivalent to a channel without the known-noise.

This abstraction is important to watermarking because the channel with the two noise sources is also known as *informed watermarking*. The two noises in Fig. 2.2 can be seen as the cover work (known-noise) and the attack (unknown-noise) to watermarking, Fig. 2.3. Thus (2.1) shows analogously that the watermarking capacity does not depend on the cover work.

However, Costa's result cannot be straightforward applied to practical solutions because Costa considered the transmission of bits taken from huge and random codebooks. Nevertheless suboptimal watermarking approaches based in the same principle, e.g. quantization index modulation (QIM) watermarking, have shown good and interesting properties.

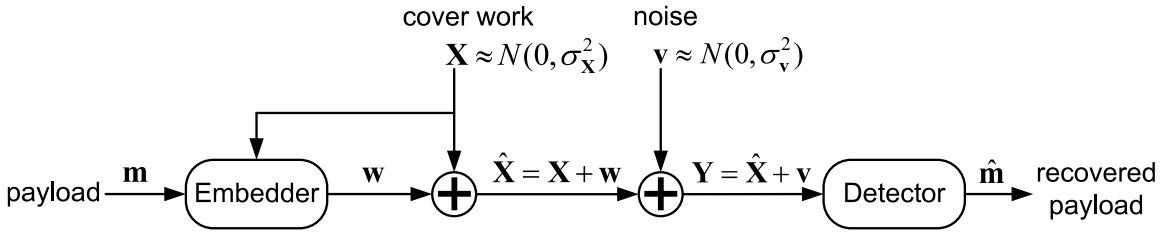


Fig. 2.3: Watermarking as communications with side information.

2.1.1 Applications of audio watermarking

The most popular application of watermarking is the insertion of a code, usually a few bits, into the digital work to resolve right ownership. However watermarking can be applied to many more applications. For example, high payload watermarking has been used to extend narrow bands from 8 kHz to 16 kHz in telephony speech [47], which results in better quality of the voice with the same physical resources.

Common applications of audio watermarking are:

- Ownership protection. The watermark enables the owner to demonstrate the presence of this watermark in case of dispute of ownership.

- Authentication and tampering detection. The watermark is used to determine whether the cover work was tampered.

- Fingerprinting. Additional data embedded by a watermark in the fingerprinting applications are used to trace the originator or recipients of a particular copy of a multimedia file.

- Broadcast monitoring. Watermarking is an obvious alternative method of coding identification information for an active broadcast monitoring. It has the advantage of being embedded within the cover work rather than exploiting a particular segment of the broadcast signal.

- Copy control and access control. The embedded watermark represents a certain copy control or access control policy. A watermark detector is usually integrated in a recording or playback system.

- Information carrier. High embedding capacity is desirable in this application. While the robustness against intentional attacks is not required, however a certain degree of robustness against processing like MPEG compression may be desirable. The watermark can be used as link to external data bases, copyright information, licensing conditions or metadata information.

The algorithms presented in this dissertation are suitable to *information carrier* applications. Our achieved payloads are high, around 200 bps, and they are robust to MPEG-1 compression. This means that the watermark can carry with at least 25 characters (assuming 8 bits/character) per second, perfectly fitting in applications like advertising or metadata information.

2.2 Watermarking is not cryptography

Many comparisons have been made between watermarking and cryptography since the interest on the former is strongly motivated by security on multimedia. However both them have different goals but they can complement each other. Cryptography is related with *secure transmission* involving privacy, integrity and authentication of the data. Meanwhile, watermarking concerns with *reliable transmission*, Fig. 2.4.

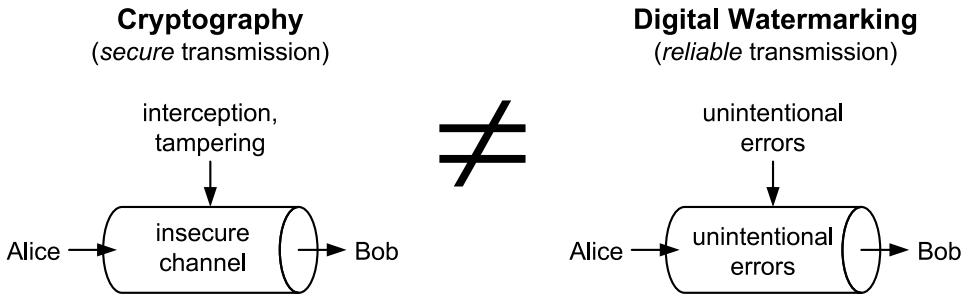


Fig. 2.4: Concerns of cryptography and watermarking.

Given the watermarking objective, why watermarking is considered as an analogy to copyrights protection?. Well, copyright protection is only an *application of watermarking* which has become very popular due to the social impact of its realization. The most basic requirement of watermarking is the ability to embed and detect a hidden message within a host signal. However, *applications of watermarking*, e.g. authentication, fingerprinting, etc., might require very much more. Therefore security threats depend on the *application of watermarking* which must not be confused with watermarking itself.

Secure systems can be built using different layers, similar to Fig. 2.5. In such design, watermarking is responsible for the synchronization and delivery of bits while cryptography is responsible of guaranteeing its privacy, integrity and authenticity.

In this dissertation, watermarking is assumed to be the responsible only of the *information hiding layer* in a secure system.

2.3 Quantization index modulation

In 2001 Chen and Wornell introduced a new watermarking algorithm, called quantization index modulation (QIM) [21], based on Costa's abstraction. The QIM foundation is to embed

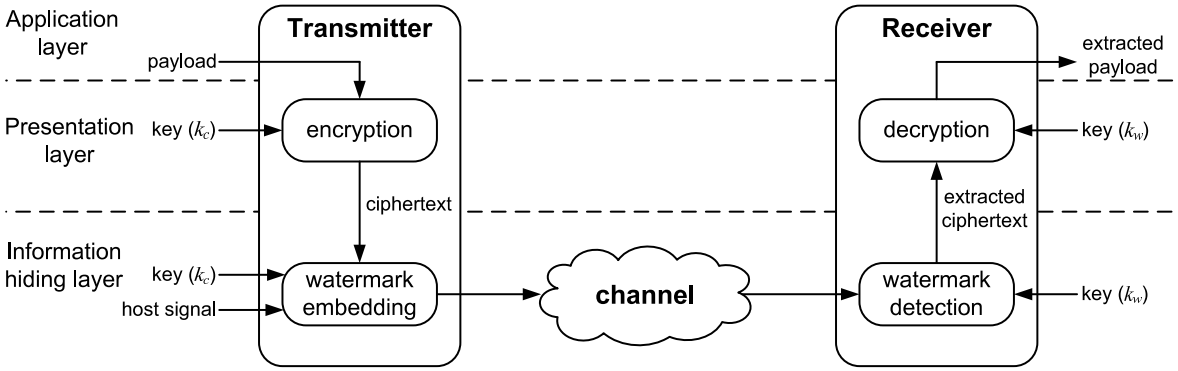


Fig. 2.5: Design of secure systems using layers.

the watermark by means of quantizing the cover work. Thereby QIM is easy to implement in software or hardware, has low complexity, and is amenable to theoretical analysis.

Let us assume that the watermark is a binary vector $\mathbf{m} = m_1, m_2, \ldots$ where $m_i \in \{0, 1\}$. Furthermore, the cover work is a vector $\mathbf{X} = X_1, X_2, \ldots$ with zero mean and variance $\sigma^2_{\mathbf{X}}$. Along this dissertation we use bold letters, e.g. $\mathbf{X}$ or $\mathbf{x}$, to represent vectors and X_i or x_i refer to their respective i-th element.

QIM associates each symbol of the watermark with a different quantizer, that is, $m_i = 0$ is related to the quantizer Q_0 and $m_i = 1$ associated to Q_1. Thus, QIM embeds the watermark by quantizing the cover work with the appropriate quantizer using the next function:

$$\hat{X}_i = Q_j(X_i, \Delta), \quad \text{for } j = m_i;$$

where Δ is the step quantization size and $\hat{\mathbf{X}} = \hat{X}_1, \hat{X}_2, \ldots$ is the watermarked work. The step quantization size is directly related with the trade-off between robustness and imperceptibility of the watermark. A bigger Δ produces more robustness but also higher degradation of the audio quality.

In the watermark extraction is assumed that a noisy version $\mathbf{Y} = \hat{\mathbf{X}} + \mathbf{v}$ of the watermarked work is received, where $\mathbf{v}$ is the noise. Then, the extracted watermark $\hat{m}_i$ is defined by computing the distance between the received samples Y_i and its quantization $Q_j(Y_i, \Delta)$. Finally, the extracted bit is equal to the argument of j which minimize the distance:

$$\hat{m}_i = \arg\min_j |Y_i - Q_j(Y_i, \Delta)|, \quad \text{for } j \in \{0, 1\}.$$

2.3.1 Dither modulation

Dither modulation (DM) is a practical implementation of QIM based on scalar and uniform quantizers which produces a very low-complex embedding and extracting algorithms.

DM uses only one quantizer Q to embed the watermark, however two constants known

as *dithered parameters* are needed. The first dithered parameter $v(0)$ is generated in pseudo-random way with a uniform distribution between $[-\Delta/2, \Delta/2]$, where Δ is the quantization step size. The second parameter $v(1)$ is computed according to:

$$v(1) = \begin{cases} v(0) + (\Delta/2), & \text{if } v(0) < 0 \\ v(0) - (\Delta/2), & \text{if } v(0) \geq 0 \end{cases}.$$

In that way, it is ensured that both dithered parameters differ in $\Delta/2$ each other.

Unlike QIM, DM does not associate the watermark with the quantizer but with the dithered parameters. Thereby, $m_i = 0$ and $m_i = 1$ are associated with $v(0)$ and $v(1)$ respectively. DM embeds the watermark m_i by adding the corresponding dithered parameter $v(m_i)$ to the digital work X_i, quantizing the result with Q, and then subtracting the same dithered parameter $v(m_i)$ added in the beginning. This process is summarized in the next function:

$$\hat{X}_i = Q\big(X_i + v(m_i), \Delta\big) - v(m_i), \tag{2.2}$$

where $\hat{\mathbf{X}} = \hat{X}_1, \hat{X}_2 \dots$ is the watermarked work.

The watermark extraction follows the same foundations than QIM which is the measurement of distances between the received signal $\mathbf{Y} = Y_1, Y_2, \dots$ and its closest reconstruction points. The recovered watermark, $\hat{m}_i$, is the argument of j which minimizes:

$$\hat{m}_i = \arg\min_j |Y_i - Q\big(Y_i + v(j), \Delta\big) + v(j)|, \quad \text{for } j \in \{0, 1\}. \tag{2.3}$$

2.3.2 Spread-Transform dither modulation

Spread-transform watermarking is based on the idea that the watermark is not directly embedded into the original work $\mathbf{X} = X_1, X_2, \dots$ but into the projection $\mathbf{S}$ of a set of the original work $\mathbf{X}_k = X_1, X_2, \dots, X_l$ onto a pseudo-random sequence $\mathbf{z} = z_1, z_2, \dots, z_l$. Although $\mathbf{z}$ is random, proper normalization must be assumed.

Let us l denotes the spreading factor, meaning the number of elements of $\mathbf{X}$ belonging to one element S_k. The term *spreading factor* is because the information to be embedded in S_k will be spread on l elements of $\mathbf{X}$ by the inverse spread-transform, thus l controls the embedding rate. Projected samples $\mathbf{S} = S_1, S_2, \dots$ are computed with

$$S_k = \sum_{i=l(k-1)+1}^{lk} X_i z_i. \tag{2.4}$$

If (2.2) is used to embed the watermark m_i into S_k, the method is called spread-transform dither modulation, denoted with ST for short. Once the bit is already embedded on S_k, the

inverse spread-transform,

$$\hat{\mathbf{X}}_k = \mathbf{X}_k - S_k\mathbf{z} + \hat{S}_k\mathbf{z}, \tag{2.5}$$

is applied to obtain the watermarked work $\hat{\mathbf{X}}$, where $\hat{S}_k$ refers to the watermarked samples in the transform domain.

In the detection phase, the received samples $\mathbf{Y}$ has to be projected onto $\mathbf{z}$ as mentioned above and then the detection method using (2.3) is applied.

2.3.3 Distortion-Compensation

Distortion-compensation (DC) is not an embedding algorithm like DM or ST, however DC is a process that improves the trade-off between distortion-robustness of QIM-based algorithms, e.g. DM and ST. The main goal of DC is to increase the reliability of the decoded bits against a certain attack by maximizing the quantization step size until a desirable level of quality degradation in the cover work.

Given a quantizer ensemble, DC scales all the quantizers by $0 < \alpha \leq 1$. Thus the square minimum distance, d^2_{min}, between reconstruction points will be increased by a factor of $1/\alpha^2$ and therefore the robustness is increased as well. Increasing d^2_{min} produces higher distortion in the cover work, hence DC compensates this additional distortion by adding back a fraction $1-\alpha$ of the quantization error to the quantization value [21]. The embedding function of QIM with DC property is:

$$\hat{X}_i = Q_j(X_i, \Delta/\alpha) + (1-\alpha)\big[X_i - Q_j(X_i, \Delta/\alpha)\big], \quad \text{for } j = m_i. \tag{2.6}$$

The second term in (2.6) is the compensation term and can be treated as independent noise at the decoder because it is statistically independent of $\mathbf{m}$. Therefore the watermark extraction process is the same as QIM.

The optimal choice for the parameter α in ideal cases depends on the watermark-to-noise power ratio (WNR) and is given by

$$\alpha_{opt} = \frac{\sigma^2_{\mathbf{w}}}{\sigma^2_{\mathbf{w}} + \sigma^2_{\mathbf{v}}} = \frac{1}{1 + 10^{-\mathrm{WNR[dB]}/10}}, \tag{2.7}$$

where $\sigma^2_{\mathbf{v}}$ is the noise power and $\sigma^2_{\mathbf{w}}$ is the watermark signal power, $\mathbf{w} = \hat{\mathbf{X}} - \mathbf{X}$.

WNR is defined as the ratio between the watermark signal power $\mathbf{w}$ and that of the noise introduced by the attack $\mathbf{v}$:

$$\mathrm{WNR} = 10\log_{10}\Big(\frac{\sigma^2_{\mathbf{w}}}{\sigma^2_{\mathbf{v}}}\Big), \tag{2.8}$$

and it is related with the strength of the attack. Along this dissertation, we will also use the

signal-to-noise ratio,

$$\mathrm{SNR} = 10 \log_{10}\Big(\frac{\sigma^2_{\mathbf{X}}}{\sigma^2_{\mathbf{w}}}\Big), \tag{2.9}$$

however it is used as measurement of the watermarked work quality.

2.4 Digital audio

The sound is the human response to certain vibrations that travel along different mediums, being air the most common one, in form of waves.

Wavelength, period, and amplitude are the main characteristics of waves. The wavelength is the horizontal distance between to equivalent points of the wave, e.g. two peaks. The time taken by the wave to complete a full cycle is called period and finally the amplitude is the height of the wave. These characteristics are pictured in Fig. 2.6. Thus, the frequency of a wave is defined as the inverse of its period, that is, the number of cycles per second.

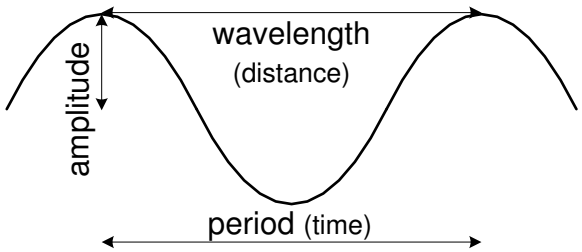


Fig. 2.6: Characteristics of waves.

For humans, hearing is normally limited to frequencies between 20 Hz and 20 kHz, although the exact limits are slightly different for each person and their age. In addition, the ear has nonlinear response to sounds of different loudness levels. This fact is used in compression techniques as well as in watermarking.

The waves of a sound can be converted in electrical signals, known as analog audio, by means of transducers, e.g. microphone. Such signals are two dimensional and they represent the variation of voltage with respect to time.

Digital audio is simply an alternative means of carrying an audio waveform. Although there are many techniques to convert analog audio into digital one, pulse code modulation (PCM) has virtually a universal use. PCM represents the audio signal as a discrete vector which contains the voltage value of the signal in certain time, Fig. 2.7. The voltage values are measured in equally intervals of time, this process is known as sampling and how frequent the samples are taken is called sampling rate. The last step of PCM is to decide how many bits are used to store the voltage value or, in other words, the accuracy of the sample.

Waveform audio (WAV) file format is a Microsoft and IBM audio format standard for storing an audio bit stream. Though a WAV file can hold compressed audio, the most com-

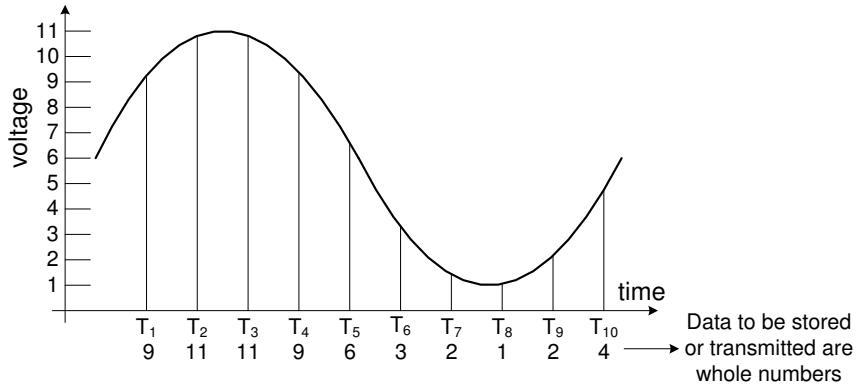


Fig. 2.7: PCM converts analog audio signals into digital ones.

mon WAV format contains uncompressed audio in PCM format. This dissertation proposes watermark algorithms for audio files in WAV format sampled at 44 100 samples per second and 16 bits per sample which is equivalent to the quality of compact discs.

2.4.1 Audio watermarking singularity

Watermarking of audio signals is more challenging compared to watermarking of images or video sequences due to wider dynamic range of the human auditory system (HAS) in comparison with the human visual system. The HAS perceives sounds over a range of power greater than $10^9 : 1$ and a range of frequencies greater than $10^3 : 1$. The sensitivity of HAS to the additive white Gaussian noise is high as well; this noise in a sound file can be detected as low as 70 dB below ambient level. On the other hand, opposite to its large dynamic range, HAS contains a fairly small differential range. That is, loud sounds generally tend to mask out weaker sounds. Additionally, HAS is insensitive to a constant relative phase shift in a stationary audio signal and some spectral distortions interprets as natural, perceptually non-annoying ones [48].

Auditory perception is based on the critical band analysis in the inner ear where a frequency-to-location transformation takes place along the basilar membrane. The power spectra of the received sounds are not represented on a linear frequency scale but on limited frequency bands called critical bands. The auditory system is usually modeled as a band-pass filter bank, consisting of strongly overlapping bandpass filters with bandwidths around 100 Hz for bands with a central frequency below 500 Hz and up to 5000 Hz for bands placed at high frequencies. If the highest frequency is limited to 24000 Hz, 26 critical bands have to be taken into account.

Two properties of the HAS used by watermarking algorithms are frequency masking and temporal masking. This concept was firstly used in wideband audio coding. Masking properties are used to embed additional bits into an existing bit stream without generating

audible noise in the audio sequence.

The requirements of watermarking in audio are inaudibility, robustness to attacks, and the embedding capacity. When one of them is increased the remaining two are affected. Therefore is very difficult to obtain a high payload algorithms with good robustness.

2.5 Attacks

In digital watermarking any modification of the watermarked work is considered as *attack*. The modification can be unintentionally produced by common audio processing or by an attacker aiming to break the watermark. In either case the watermark must be fully or partially recovered depending of the application, if and only if the watermarked work still has enough quality to hold commercial value.

This dissertation proposes algorithms tolerant to compression and desynchronization. The former is conveyed by using MP3 compression algorithm which is considered a tough attack to audio watermarking. The latter, desynchronization, is considered still an open problem to watermarking.

2.5.1 MPEG-1 layer III (MP3)

MPEG-1 Audio is a popular audio encoding and lossy compression format, designed to greatly reduce the amount of data required to present audio, yet still sound like a faithful reproduction of the original uncompressed audio to most listeners [49].

The MPEG-1 standard incorporates several methods including subband decomposition, filter-bank analysis, transform coding, entropy coding, dynamic bit allocation, nonuniform quantization, adaptive segmentation, and psychoacoustic analysis. The compression algorithm operates on 16-bit PCM input data at samples rates of 32, 44.1 and 48 kHz. Moreover MPEG-1 offers separate modes for mono, stereo, dual independent mono, and joint stereo.

The MPEG-1 architecture contains three layers of increasing complexity, delay, and output quality. Each higher layer incorporates functional blocks from the lower layers. The layer III is the most efficient in the sense of audio-quality versus compression-rate, thus is the most popular one and it is known as MP3.

MP3 architecture, Fig. 2.8, achieves performance improvements by adding several important mechanisms on top of the layer I/II foundation. MP3 algorithm operates on consecutive frames of data. Each frame consists of 1152 audio samples; a frame is further split into two subframes of 576 samples each. A subframe is called granule. At the decoder, every granule can be decoded independently. A hybrid filter bank is introduced to increase frequency resolution and thereby better approximate critical band behavior. The hybrid filter bank includes adaptive segmentation to improve pre-echo control. Sophisticated bit allocation and

quantization strategies that rely upon nonuniform quantization, analysis-by-synthesis, and entropy coding are introduced to allow reduced bit rates and improved quality. The hybrid filter bank is constructed by following each subband filter with and adaptive modified discrete cosine transform. This practice allows for higher-frequency resolution and pre-echo control. Shorter blocks (4 ms) provide for temporal pre-masking and pre-echoes during transients; longer blocks during steady-state periods improve coding gain by reducing side information and hence bit rate.

Bit allocation and quantization of the spectral lines are realized in a nested loop procedure that uses both nonuniform quantization and Huffman coding. The inner loop adjust the nonuniform quantizer step sizes for each block until the number of bits required to encode the transform components falls within the bit budget. The outer loop evaluates the quality of the coded signal in terms of quantization noise relative to the *just noticeable difference* (JND) thresholds [50].

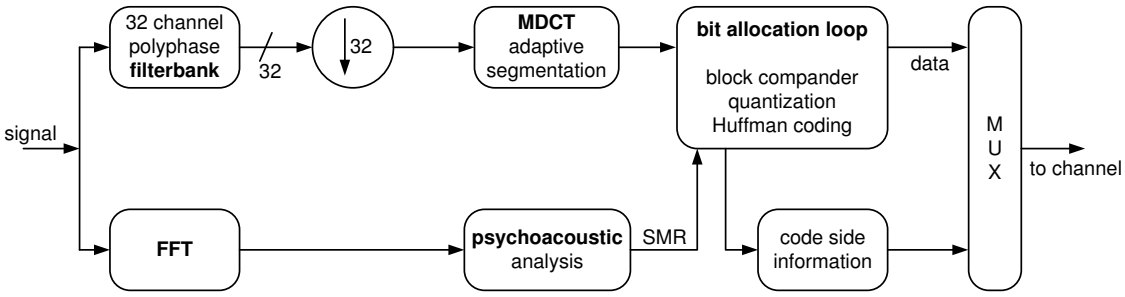


Fig. 2.8: ISO/MPEG-1 layer III (MP3) encoder.

2.5.2 Clipping attack

Most of the watermarking schemes assume perfect synchrony, that is, both embedder and detector should know the correct place where the watermark is embedded. Moreover, if the information was divided and embedded as blocks of information due to any error-control coding technique then the detector must be capable to detect the beginning of each block.

Real applications are susceptible to suffer desynchronization attacks and therefore without a synchronization scheme the detector might emits unintelligible results even in noise free channels.

Clipping attack attempts to desynchronize the detector by removing or clipping random samples along the cover work. Fig. 2.9 pictures this situation. Each square represents a sample of the cover work, solid-line squares are where the watermark was embedded. Since samples 2, 6 and 13 were removed, the detector faces problems to distinguish the beginning of the watermark as well the bounds of each block of information.

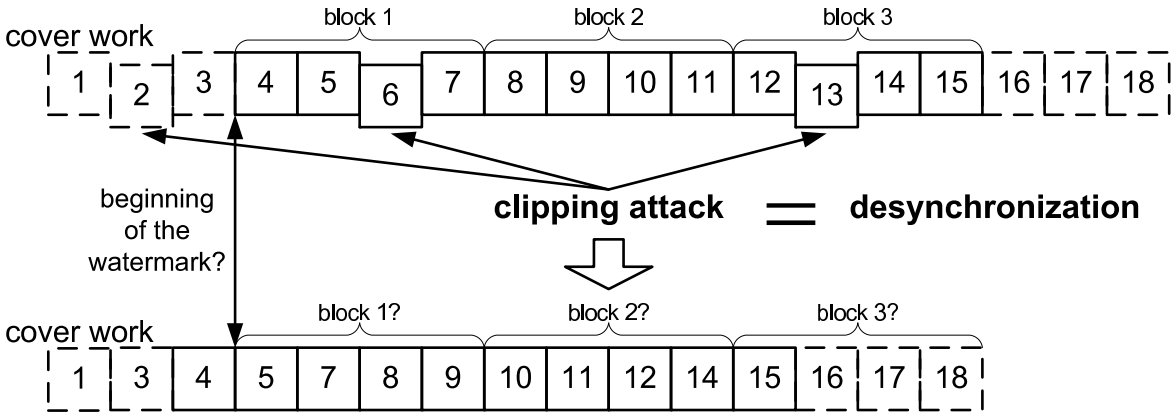


Fig. 2.9: Clipping attack removes random samples from the cover work.

2.6 Low-density parity-check codes

Low-density parity-check (LDPC) codes are linear codes that are constructed by designing a sparse parity check matrix H. They were discovered by Robert Gallager in 1962 [4], however because the computational complexity of its decoding algorithm were forgotten until early 80's when Tanner introduced an idea to represent codes based on graphs. But it was not until 1996 when LDPC codes took great attention due to MacKay and Neal showed that LDPC could perform as well as turbo codes on the additive white Gaussian noise (AWGN) channel [12].

The good properties of LDPC codes arise when the code length is increased because the error probability decreases exponentially, its minimum distance is increased, and the complexity grows linearly.

2.6.1 Construction

LDPC codes can be defined only with its parity check matrix H. The generator matrix G can be obtained from H. Gallager proposed an LDPC construction based in pseudo-random allocation of the ones along H. The construction is the following:

- Make a sparse parity check matrix H by randomly determining the position of 1's with a fixed number, s and v, of 1's per column and per row respectively.
- The number of 1's per column must satisfies $s > 2$
- The overlapping of 1's per column and per row should be at most equal to one.
- s and v should be small number compared with the code length n.

The above construction often produces H in non-systematic form. Therefore, using Gaussian elimination H can be converted to systematic form $H' = [I_{n-k}\ P^T]$, where n is the

code length, k the information block length and I_{n-k} is an identity submatrix of dimension $(n-k) \times (n-k)$. Finally we can obtain the generator matrix $G = [P\ I_k]$.

In 1981 Tanner [5] proposed and pictorial representation of the parity check matrix using a bipartite graph. Tanner graph, named after its inventor, depicts the relations between symbols nodes which represent the transmitted symbols and the parity checks nodes which represent the parity check equations. Given an LDPC code, if the entry $\{i, j\}$ of the sparse parity check matrix H is equal to one, $H_{i,j} = 1$, then there is connection between the symbol node j and the check node i. For example, assume the next parity check matrix H of a Hamming code $(7, 4)$,

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix},$$

thus the corresponding Tanner graph is show in Fig. 2.10.

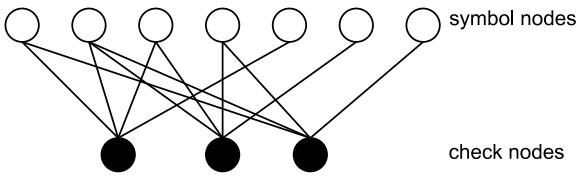


Fig. 2.10: Tanner graph of the parity-check matrix H of the Hamming code $(7, 4)$.

2.6.2 Encoding

Assuming that the information is represented by a binary vector $\mathbf{m}$, the encoding is produced by the binary multiplication between $\mathbf{m}$ and the generator matrix G:

$$\bar{\mathbf{m}} = \mathbf{m} \cdot G,$$

so that the coded vector $\bar{\mathbf{m}}$ satisfies the syndrome equation $\bar{\mathbf{m}} \cdot H^T = 0$, where H^T is the transpose of the parity-check matrix H.

2.6.3 Decoding: Sum-product algorithm

Decoding of LDPC codes can be conveyed in several ways, however the most popular methods are bit-flipping (BF) and sum-product algorithm (SPA). BF [51] is a hard-decision algorithm that offers a good trade-off between error performance and decoding complexity. However soft-decision algorithms can achieve better error performance than those with hard-decision. SPA [13] is a soft-decision and iterative decoding, based on belief propagation that is extremely efficient for decoding LDPC codes and yet is practically implementable.

SPA processes the received symbols iteratively to improve the reliability of each code

symbol based on the parity-check matrix H of an LDPC code. The reliability of the coded symbol can be measured by its marginal a posteriori probability, its log-likelihood ratio (LLR) or the value of its corresponding received symbol. The computed reliability measures of code symbols at the end of each decoding iteration are used as input for the next iteration. The decoding iteration process continues until a certain stopping condition is satisfied. Then, based on the computed reliability measures of code symbols, hard decisions are made [51].

Let $h_{i,j}$ denote the entry of H in the i-th row and the j-th column. Let

$$\mathcal{L}(k) = \{\ell : h_{k,\ell} = 1\}$$

denote the set of code positions that participate in the m-th parity-check equation, and let

$$\mathcal{K}(\ell) = \{k : h_{k,\ell} = 1\}$$

denote the set of check positions in which code position ℓ participates.

The SPA iteratively computes two types of conditional probabilities:

$q^x_{k\ell}$, the probability that the ℓ-th bit of $\bar{\mathbf{m}}$ has the value x, given the information obtained via the check nodes other than check node k.

$r^x_{k\ell}$, the probability that a check node k is satisfied when bit ℓ is fixed to a value x and the other bits are independent with probabilities $q_{k\ell'}, \ell' \in \mathcal{L}(k) \setminus \ell$.

Assuming binary transmission over an AWGN channel and that the modulated symbols $\phi(\bar{m}_i) = (-1)^{\bar{m}_i}\sqrt{E_s}$ are received as $y_i = \phi(\bar{m}_i) + v_i$, where v_i is a Gaussian distributed random variable with zero mean and variance $\sigma^2_{\mathbf{v}}$.

SPA is initialized as it follows. For $\ell \in \{1, 2, \ldots, N\}$, initialize the a priori probabilities of the code nodes

$$p^1_\ell = \frac{1}{1 + \exp(y_\ell \frac{2}{\sigma^2_{\mathbf{v}}})}$$

and $p^0_\ell = 1 - p^1_\ell$. For every (ℓ, k) such that $h_{k,\ell} = 1$,

$$q^0_{k,\ell} = p^0_\ell, \qquad q^1_{k,\ell} = p^1_\ell.$$

Then, the message passing of SPA consists of two steps:

Step 1: Bottom-up (horizontal)

For each ℓ, k, compute

$$\delta y_{k,\ell} = \prod_{\ell' \in \mathcal{L}(k)\setminus\ell} (q^0_{k,\ell'} - q^1_{k,\ell'}),$$

and

$$y^0_{k,\ell} = (1 + \delta y_{k,\ell})/2, \qquad y^1_{k,\ell} = (1 - \delta y_{k,\ell})/2.$$

Step 2: Top-down (vertical)

For each ℓ, k, compute

$$q^0_{k,\ell} = p^0_\ell \prod_{k' \in \mathcal{K}(\ell)\setminus k} y^0_{k',\ell}, \qquad q^1_{k\ell} = p^1_\ell \prod_{k' \in \mathcal{K}(\ell)\setminus k} y^1_{k',\ell},$$

and normalize, with $\alpha = 1/(q^0_{k\ell} + q^1_{k\ell})$,

$$q^0_{k,\ell} = \alpha q^0_{k,\ell}, \qquad q^1_{k,\ell} = \alpha q^1_{k,\ell}.$$

For each ℓ compute the a posteriori probabilities

$$q^0_\ell = p^0_\ell \prod_{k \in \mathcal{K}(\ell)} y^0_{k,\ell}, \qquad q^1_\ell = p^1_\ell \prod_{k \in \mathcal{K}(\ell)} y^1_{k,\ell},$$

and normalize, with $\alpha = 1/(q^0_\ell + q^1_\ell)$,

$$q^0_\ell = \alpha q^0_\ell, \qquad q^1_\ell = \alpha q^1_\ell.$$

Finally the soft-outputs are decoding with:

$$\hat{m}_i = \operatorname{sgn}\Big(q^0_i - \frac{1}{2}\Big),$$

for $i = 1, 2, \ldots, N$. If $\hat{m}H^\top = \bar{0}$, then $\hat{m}$ is the estimated codeword and the soft outputs are

$$\Lambda(m_i) = \log(q^1_i) - \log(q^0_i), \quad 1 \le i \le N.$$

The algorithm stops.

Otherwise, return to Step 2. If the number of iterations exceeds a predetermined threshold, a decoding failure is declared.

3

Audio watermarking

Audio watermarking started slightly later than watermarking for still images. The first proposal dedicated to audio signals was published in 1996 by Boney [10].

Early research of audio watermarking was oriented to adapt algorithms originally developed for images, e.g. *patchwork* or *spread spectrum*, to audio files.

Patchwork [9] technique separates the data to be watermarked into two distinct subsets. One feature of the data is chosen and modified in opposite directions in both subsets. The separation of the samples is the secret used in the embedding and detection step. The watermark can be easily detected if the data satisfies some statistical properties.

Spread spectrum (SS) communications were invented to send guidance information to torpedoes believed to be resistant to jamming. SS foundations were applied to digital watermarking for images by Cox [6] and later by Kirovski to audio signals [30]. In SS, each message is represented by a pseudo-random sequence which is spread along different frequencies of the signal, the sequence must be small enough to avoid serious degradation of the host signal. The detection is usually conveyed by a matched filtering.

Either patchwork or SS represents watermark algorithms with good robustness but their embedding rates are low.

High embedding rates can be achieved with quantization index modulation (QIM) algorithms, however the cost to be paid is weak robustness. Alternatives to increase the robustness are the embedding in frequency domain and addition of error-control codes (ECC) to encode the watermark.

This Chapter is dedicated to the analysis of QIM algorithms applied to audio files with the purpose of setting up a watermark framework for the Chapter 4. Our interest is to study the behavior of different QIM-based algorithms in frequency domain and its robustness against attacks.

Firstly, a brief introduction about the one-dimensional discrete wavelet transform is given in Section 3.1. In the same Section, we also describe how the frequency coefficients can

be obtained from the audio file and which of them are selected to embed and retrieve the watermark.

Then, algorithms to embed the watermark in the audio's frequencies using dither modulation (DM) and spread-transform dither modulation (ST) are explained in Section 3.2.

Watermarking channels tend to be very noisy and QIM alone is not enough to obtain robust watermarks even if the watermarks are embedded in frequency. Therefore, we propose the addition of powerful codes like low-density parity-check (LDPC) codes to encode the watermark and thus improving the weakness of QIM. Section 3.3 addresses the implementation of QIM-based algorithms with LDPC codes and Section 3.4 shows comparisons and analysis of all watermark algorithms, including coded and uncoded versions. We compare the robustness of the algorithms against Gaussian noise and common audio processing techniques.

Finally the last Section, 3.5, of this Chapter is regarding small improvements which can be implemented in practical systems but they are not deeply studied because are out of the scope of this dissertation.

3.1 Embedding on frequency domain

The embedding in frequency domain emerges due to the need of robust watermarks. Frequency is more robust than time domain because the watermark is placed in the fundamental frequencies of the audio signal. Thus, the attacker should introduce strong distortions to the audio, which makes the audio worthless, in order to destroy or remove the watermark.

The frequency representation of a signal can be computed with mathematical transformations like fast Fourier transform (FFT), discrete cosine transform (DCT), and discrete wavelet transform (DWT). Each of them, depending of the signal to be analyzed, represents different constraints of complexity and frequency response. In our algorithms, DWT is used to compute the frequency representation of the audio signals.

3.1.1 Discrete wavelet transform

Wavelet is a mathematical transform that provides a time-frequency representation. Unlike Fourier transform, wavelet produces more accurate information about the frequencies and when they occur, especially in non-stationary signals like audio files. Although the wavelet natural representation is based on scale-time, the representation of the information in frequency-time is straight forward.

DWT offers decomposition oriented to low-frequencies. In other words, if we picture the wavelet decomposition as a binary tree, Fig. 3.1, where the left node belongs to low-frequencies and right node to high-frequencies then, the decomposition is done always to the left node until the desired level.

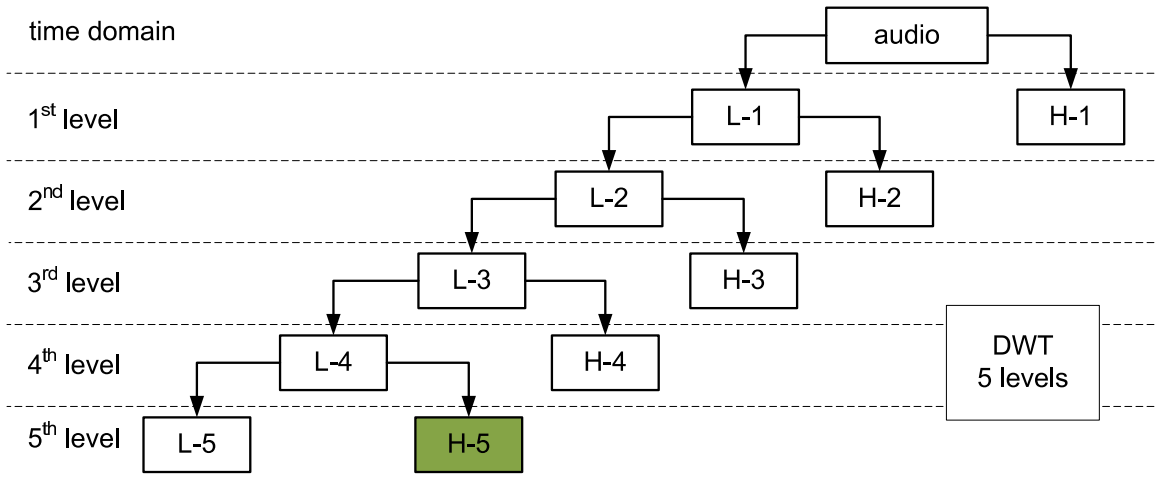


Fig. 3.1: Decomposition of the audio file using discrete wavelet transform.

Thereby the lowest left node, in this example "L-5" of Fig. 3.1, is considered as the approximation coefficients of the signal which contain the fundamental information. The lowest right node, "H-5", is called the detail coefficients.

3.1.2 Framework of the embedding

Let us assume that the audio file is represented by a vector $\mathbf{X} = X_1, X_2, \ldots$ where each X_i denotes an audio sample in time domain. The vector $\mathbf{X}$ is divided into non-overlapping blocks of n continuous samples. DWT is applied to each block until the fifth level.

Since small changes in the approximation coefficients, "L-5", produce audible distortion in the audio file, only coefficients from the block "H-5" are taken to form the vector $\mathbf{x} = x_1, x_2, \ldots$. Other frequency coefficients are kept for later reconstruction of the audio file.

The watermark $\mathbf{m} = m_1, m_2, \ldots$ is embedded in each coefficient of $\mathbf{x}$ using the embedding function $f(x_i, m_i) = \hat{x}_i$, producing the watermarked coefficients $\hat{\mathbf{x}} = \hat{x}_1, \hat{x}_2, \ldots$.

The watermarked audio $\hat{\mathbf{X}} = \hat{X}_1, \hat{X}_2, \ldots$ is reconstructed using the blocks L-5, H-4, H-3, H-2, H-1 and $\hat{\mathbf{x}}$ by means of the inverse discrete wavelet transform (IDWT). This process is summarized in Fig. 3.2 and will be assumed for all the algorithms described in this Chapter.

3.1.3 Framework of the detection

At the detection phase the attacked audio $\mathbf{Y} = \hat{\mathbf{X}} + \mathbf{v}$ is assumed to be received, where $\mathbf{v}$ is noise and $\hat{\mathbf{X}}$ is the watermarked audio.

The audio file $\mathbf{Y}$ is divided in blocks of n samples, similar to the embedding phase. DWT until the fifth level is applied to each block and only the detail coefficients, "H-5", are used.

The detection function $g(y_i)$ takes the frequency coefficients $\mathbf{y} = y_1, y_2, \ldots$ and decides the bit of the recovered watermark $\hat{\mathbf{m}} = \hat{m}_1, \hat{m}_2, \ldots$. Fig. 3.3 summarizes the detection

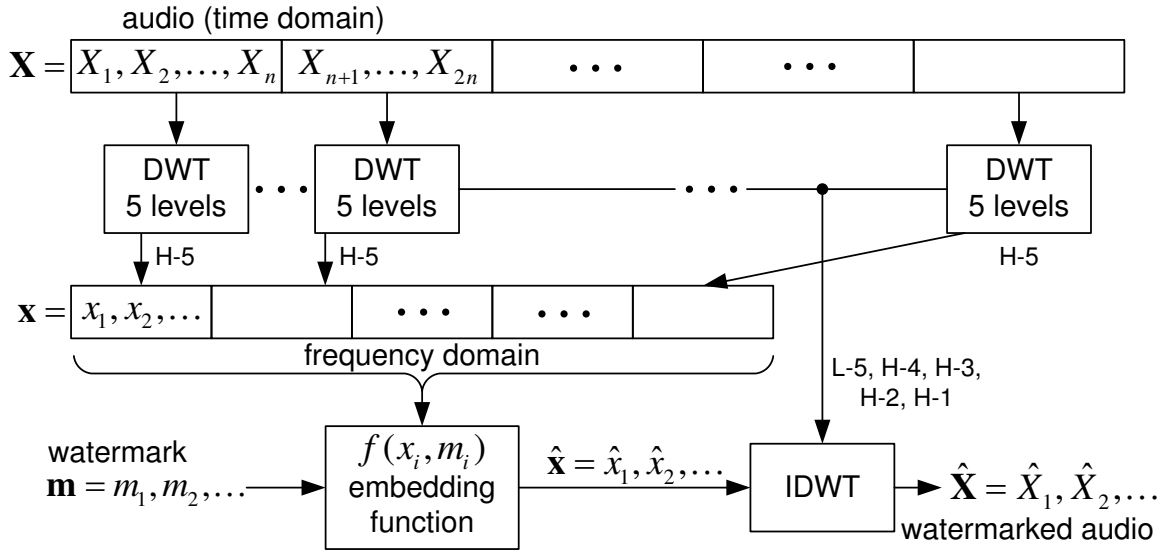


Fig. 3.2: General embedding scheme on frequency domain for audio files.

structure.

This framework will be also used in the watermark algorithms described in this Chapter. The objective is to compare the performance of different embedding functions like DM or ST but using the same embedding and detection framework.

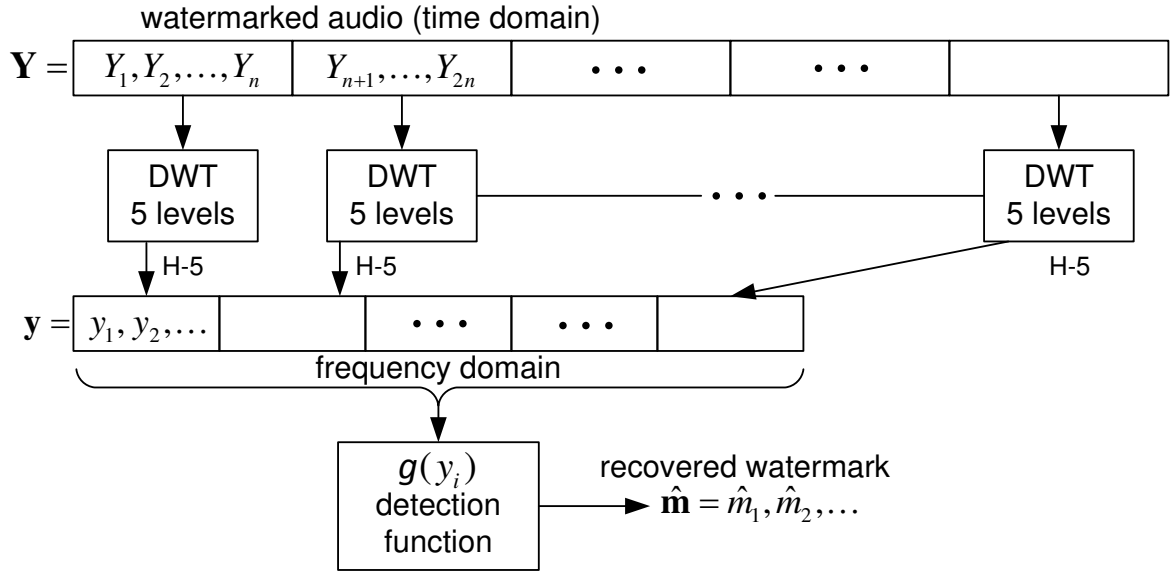


Fig. 3.3: General detection scheme on frequency domain for audio files.

3.2 QIM-based watermarking

Audio watermarking schemes based on dither modulation (DM) and spread-transform dither modulation (ST) will be proposed and analyzed in this Section. Inside the analysis, we also include improvements of DM and ST by using distortion-compensation (DC).

Each algorithm will be compared by measuring its bit error rate (BER) over a channel contaminated with AWGN. In order to obtain fair comparison, the step size $\Delta = .02$ and the embedding rate ≈ 20 bps is the same to all the schemes.

The audio files used in the simulations are in WAVE format, sampled at 44.1 kHz with 16 bits, and monaural.

3.2.1 Dither modulation

DM is the simplest watermark algorithm based on QIM. The watermark is embedded following the idea pictured in Fig. 3.2 with the characteristic that the embedding function $f(x_i, m_i)$ is given by DM.

The audio file is divided in blocks of 512 samples, for each block DWT with *Haar* mother-wavelet and five levels are computed. Only detail coefficients $\mathbf{x} = x_1, x_2, \ldots$ from the fifth level are chosen for embedding.

Even the embedding is conveyed in frequency domain, the watermark is still weak. Therefore the watermark has to be embedded repeatedly in order to increase its robustness. This redundancy can be controlled by coding the watermark with a repetition code before the embedding, Fig. 3.4.

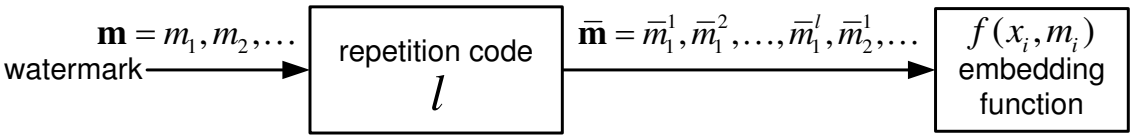


Fig. 3.4: Using repetition codes before embedding

For example, Fig. 3.5 shows the performance of DM using different repetition codes of length l. The horizontal axis measures the robustness using WNR, (2.8), the more left the better robustness. The robustness is clearly improved when the repetition code is increased. From now and in further Sections and Chapters, we will refer to this method as the "uncoded DM" even it uses repetition codes.

Thinking about the payload-robustness constraint, we must define an appropriate length l for the repetition code. According to [52], the watermark rate for audio watermarking should be higher than 20 bps, then fixing $l = 68$ our proposal obtains a watermark rate of 20.26 bps. After the repetition code, each symbol from the coded watermark $\bar{\mathbf{m}}$ is embedded in $\mathbf{x}$ using the function (2.2).

Reconstruction of the watermarked audio file $\hat{\mathbf{X}}$ is conveyed by the IDWT using the watermarked coefficients $\hat{\mathbf{x}}$ and the unused coefficients as showed in Fig. 3.2.

Watermark detection proceeds as in Fig. 3.3 until obtain the watermarked coefficients $\mathbf{y} = y_1, y_2, \ldots$. In order to decide whether zero or one was embedded in y_i, the distance d_i is

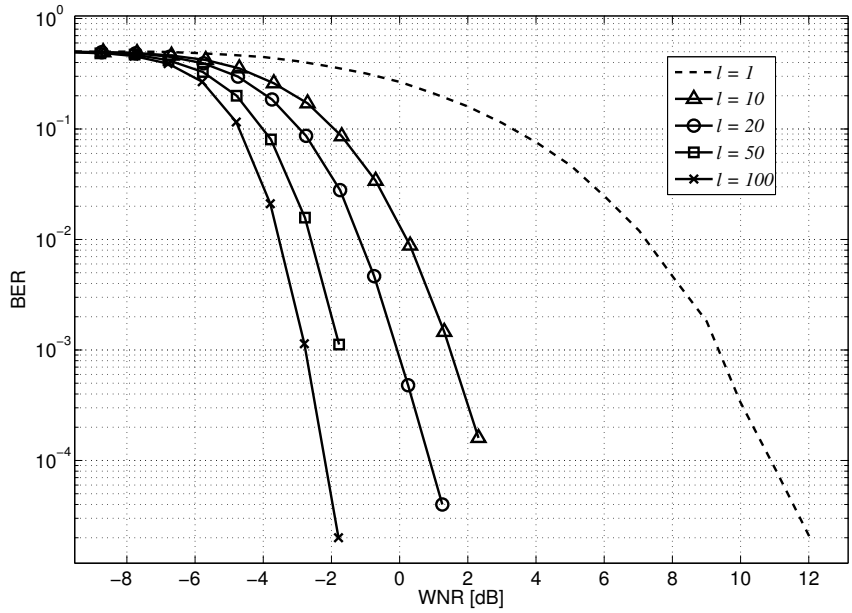


Fig. 3.5: DM robustness using repetition codes with different length l.

computed between $\mathbf{y}$ and its quantization with respect to zero:

$$d_i = |y_i - Q(y_i + v(0), \Delta) + v(0)|. \tag{3.1}$$

The result from (3.1) still contains the redundancy added by the repetition code and its mean is different from zero. Then, subtracting $\Delta/4$ from each symbol we obtain zero mean and the repetition code can be easily decoded by adding the contribution from each symbol,

$$\hat{\hat{m}}_k = \sum_{i=l(k-1)+1}^{lk} d_i - \frac{\Delta}{4}.$$

Finally the recovered watermark $\hat{\mathbf{m}} = \hat{m}_1, \hat{m}_2, \ldots$ is decided according to:

$$\hat{m}_k = \begin{cases} 0, & \text{if } \hat{\hat{m}}_k < 0, \quad \text{or} \\ 1, & \text{if } \hat{\hat{m}}_k \geq 0 \end{cases}.$$

The performance of this proposal, which uses DM, is showed in Fig. 3.6 with dotted line. DM obtains a robustness of WNR $= -14$ dB at BER $= 10^{-4}$ against AWGN.

The performance of DM can be improved with a post-processing called DC. This improvement is based on increasing the minimum distance between reconstruction points and, at the same time, controlling the distortion due to the increment. The embedding framework is the same as DM however the embedding function is changed to:

$$\hat{x}_i = \Big\{ Q\big(x_i + v(j), \frac{\Delta}{\alpha}\big) + (1-\alpha)\Big[x_i - Q\big(x_i + v(j), \frac{\Delta}{\alpha}\big)\Big] \Big\} - v(j), \tag{3.2}$$

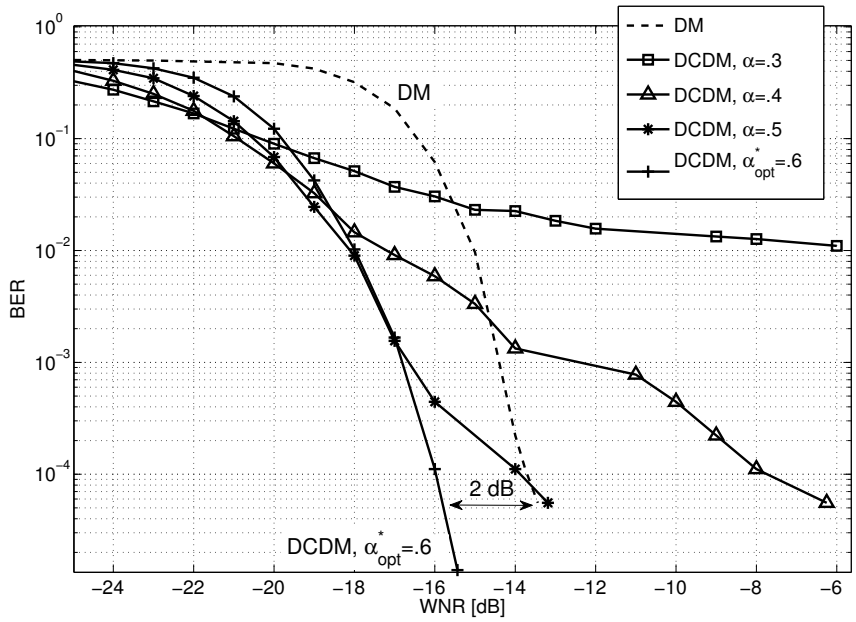


Fig. 3.6: Performance of DM and DCDM over AWGN channel.

where α is a parameter that scales Δ. We refer to this algorithm as distortion-compensated dither modulation (DCDM).

In the ideal case, the optimal value of α is computed with (2.7). Nevertheless the ideal case is not practical because involves huge and random quantization codebooks. For suboptimal codebooks, e.g. the scalar and uniform codebook of DM, the optimal value of α^*_{opt} can be computed with [53]:

$$\alpha^*_{opt} = \frac{\sigma_{\mathbf{w}}\sqrt{12}}{\Delta}. \tag{3.3}$$

The side effect produced by the scaling of Δ is taken as additive noise at the decoder. Therefore the watermark detection of DCDM is exactly the same as DM.

DCDM's performance against AWGN is shown in Fig. 3.6. Clearly DCDM with α^*_{opt} has better performance than DM, about 2 dB better. In the same simulation we also included DCDM with $\alpha \neq \alpha^*_{opt}$. When α is far from α^*_{opt} the performance is definitely worst than DM but an interesting result is obtained with $\alpha = .5$ because it obtains better performance in low WNR than DCDM with α^*_{opt}.

The audio quality of the watermarked work was measured using the SNR defined by (2.9). DM and STDM schemes obtained an audio quality of SNR > 33 dB.

3.2.2 Spread-Transform dither modulation

Spread-Transform dither modulation (ST) is a watermark algorithm that combines properties of spread spectrum communications with QIM-based methods. While DM embeds the watermark directly into the cover work, ST embeds the watermark into a projection of the

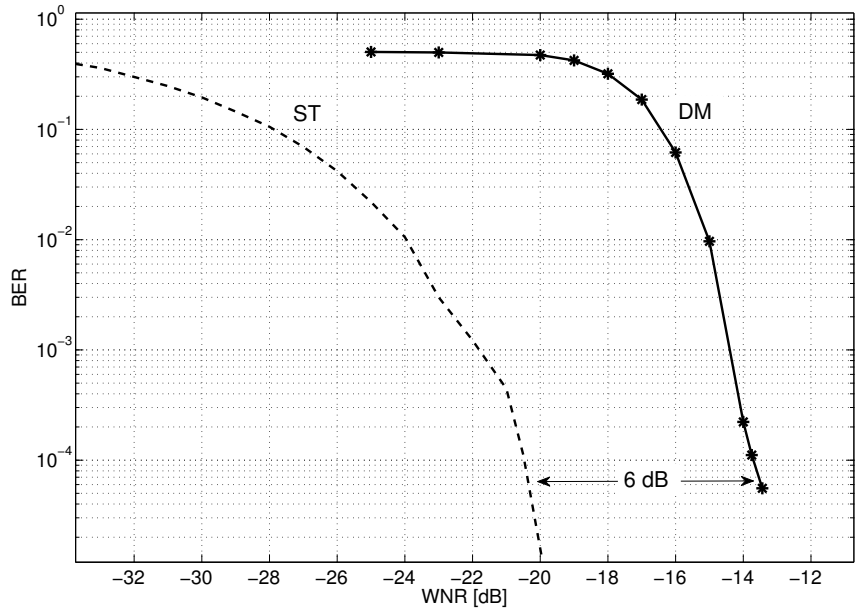


Fig. 3.7: Comparison between DM and ST audio watermarking.

cover work.

The embedding phase follows the general approach explained in Section 3.1.2 until the wavelet coefficients $\mathbf{x} = x_1, x_2, \ldots$ are obtained. The vector $\mathbf{x}$ is again divided in blocks of length l and every block is projected with (2.4) to obtain s_k. Each bit from the watermark is embedded into one projected coefficient $\mathbf{s} = s_1, s_2, \ldots$ by means of DM, $\hat{s}_k = Q(s_i + v(j), \Delta) - v(j)$, where $v(j)$ is a dither parameter, explained in Section 2.3.1. Then, the inverse spread transform (2.5) is applied to $\hat{\mathbf{s}} = \hat{s}_1, \hat{s}_2, \ldots$ and the watermarked coefficients $\hat{\mathbf{x}} = \hat{x}_1, \hat{x}_2, \ldots$ are obtained. Finally the audio file is reconstructed with the IDWT using $\hat{\mathbf{x}}$.

The signal $\mathbf{Y}$ is received at the detection phase. $\mathbf{Y}$ is decomposed in blocks and transformed to frequency domain. Detail coefficients $\mathbf{y}$ are again divided in blocks of length l and projected with (2.4). Since projections are computed in the encoder and the decoder, the pseudo-random vector $\mathbf{z}$ must be known in both embedding and detection phase. However, the same private key used to compute the dither parameters $v(j)$ can be again used to compute $\mathbf{z}$. The recovered watermark is retrieved by computing (2.3) with the projected coefficients $\mathbf{s}$ obtained from $\mathbf{y}$.

Fig. 3.7 shows the performance of ST using $l = 68$ over a channel with Gaussian noise. Remember that the parameter l was also used in Section 3.2.1 with DM for controlling the repetition code. However in ST, l is used to control the spreading factor. Although l has different meaning for DM and ST, in both cases l controls the embedding rate. In either ST or DM, the embedding rate is 20.26 bps and thus fairly comparison can be conveyed. ST obtains a robustness of WNR $= -20$ dB at BER $< 10^{-4}$, which is 6 dB more robust than DM.

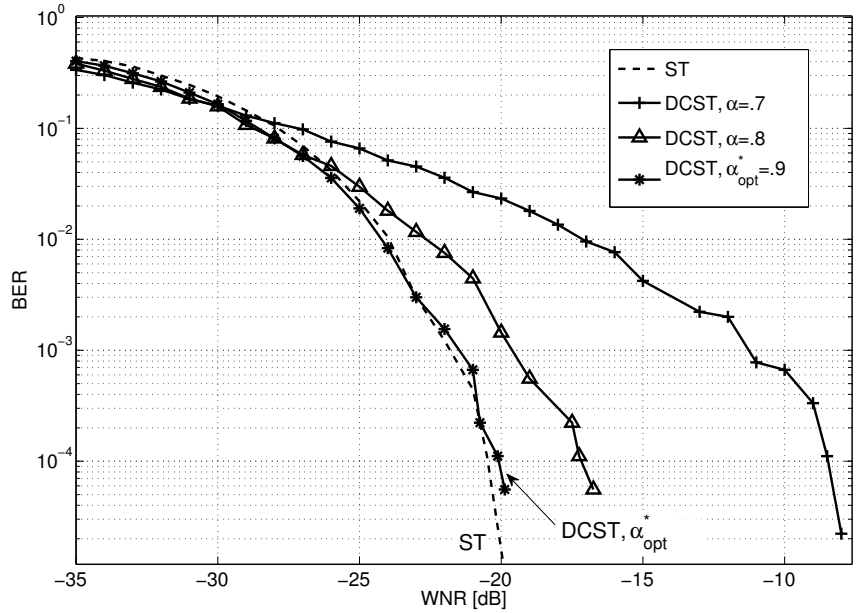


Fig. 3.8: Performance of ST and STDC over AWGN channel.

DC property can be also applied to ST and increasing its robustness. The only modification is applied in the embedding function since Δ must be scaled by α. Spread-transform dither modulation with distortion-compensation property (DCST) is defined with the next embedding function:

$$\hat{s}_i = \Big\{ Q\big(s_i + v(j), \frac{\Delta}{\alpha}\big) + (1-\alpha)\Big[s_i - Q\big(s_i + v(j), \frac{\Delta}{\alpha}\big)\Big] \Big\} - v(j). \tag{3.4}$$

When DC is applied in any quantization method, the extra distortion due to scaling of Δ is taken as additive noise at the decoder. Therefore the detection method is the same as normal ST watermarking.

Computing the scaling factor α with (3.3) and considering also $l = 68$, the optimal scale factor for this scheme is $\alpha^*_{opt} = .9$. Fig. 3.8 shows that the performance of DCST with α^*_{opt} is almost the same than ST. When $\alpha \neq \alpha^*_{opt}$, the performance of DCST decreases drastically.

According to the results of the simulations, DC property does not represent a substantial improvement for ST watermarking. On the other hand ST watermarking methods, including ST and DCST, obtain very high audio quality, SNR = 51 dB.

3.3 Increasing the watermark robustness with ECC

Watermarking is often model as the transmission of a message over a very noisy channel. Indeed, the watermark power must be very low to ensure invisibility, and the modifications suffered by the digital work can be rather strong, leading to a high noise level. To ensure a reliable transmission on a channel with such a low SNR, channel coding is thus manda-

tory [24].

In this Section we introduce improvements of the schemes of Section 3.2 by adding LDPC codes for encoding the watermark.

3.3.1 Dither modulation with LDPC codes

Using LDPC codes we expect to improve the robustness of the watermark, however coding at sampling level is tough because QIM-based algorithms are very weak. Therefore concatenation of LDPC with repetition codes is needed. Otherwise only LDPC codes cannot control the noise generated in these watermarking channels. Our general scheme for watermarking with LDPC codes is shown in Fig. 3.9.

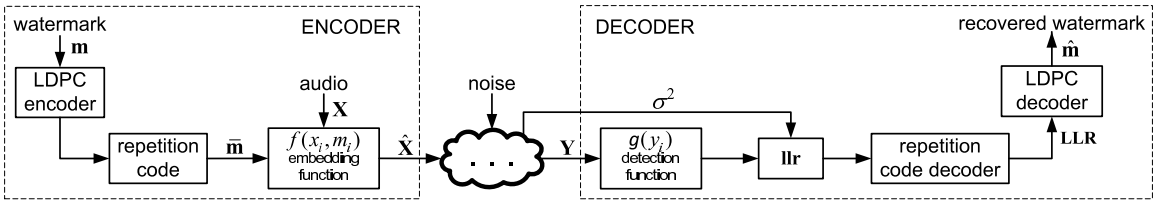


Fig. 3.9: General scheme using LDPC codes to improve the robustness in watermarking.

The watermark $\mathbf{m}$ is firstly encoded with an LDPC code and the output is again encoded with a repetition code of length l producing the coded watermark $\bar{\mathbf{m}}$. The embedding follows the same layout described in Section 3.1.2. Thus $\bar{\mathbf{m}}$ is embedded in the detail coefficients of the wavelet transform using DM.

At the detection phase, the audio $\mathbf{Y}$ is decomposed with DWT and the distance $\mathbf{d}$, between the frequency coefficients $\mathbf{y}$ and its quantization respect to zero, is measured with (3.1) in the same way as Section 3.2.1.

The soft-information $\mathbf{r} = r_1, r_2, \ldots, r_n$ is computed with $\mathbf{r} = \mathbf{d} - \Delta/4$ where Δ is the step size used by the quantizers.

LDPC decoder uses the log-likelihood ratio $\mathbf{llr} = llr_1, llr_2, \ldots, llr_n$ as metric to decode the watermark using an algorithm called the sum-product. The llr_i of a embedded bit $\bar{m}_i$ is computed as the ratio between the probability that a given value r_i could be 1 (represented by $\Delta/4$) or 0 (represented by $-\Delta/4$):

$$llr_i = \ln \frac{P(\bar{m}_i = \Delta/4 | r_i)}{P(\bar{m}_i = -\Delta/4 | r_i)}. \tag{3.5}$$

Knowledge of statistics of the channel noise is needed to compute (3.5). Using a Gaussian kernel and assuming that the statistics of the channel noise, in this case the noise variance

σ^2, are known then the **llr** can be computed with:

$$\begin{aligned} llr_i &= \ln\left(\frac{\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(r_i-(\Delta/4))^2}{2\sigma^2}\right)}{\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(r_i+(\Delta/4))^2}{2\sigma^2}\right)}\right), \\ &= \frac{\Delta r_i}{2\sigma^2}. \end{aligned} \tag{3.6}$$

The repetition code of length l is decoded with:

$$LLR_k = \sum_{i=l(k-1)+1}^{lk} llr_i. \tag{3.7}$$

Finally, $\mathbf{LLR} = LLR_1, LLR_2, \ldots$ is forwarded to the LDPC decoder and the watermark $\hat{\mathbf{m}}$ is recovered.

From now on and for the sake of simplicity, we will use the symbol " ^{c} " to discern the algorithms that use LDPC codes from those who does not; e.g. DM^c means the algorithm with dither modulation plus LDPC codes, and DM is the acronym for the uncoded dither modulation algorithm.

Recalling, the uncoded methods either DM or ST have an embedding rate of 20 bps. When the LDPC is added, we have to be aware of keeping the same embedding rate than the uncoded algorithms to produce fair comparisons. Thus, the length l of the repetition code in the coded algorithms has to be variable depending on the LDPC code. For example, let us assume that the length of the repetition code used in the uncoded DM is $l = \mathrm{L}$. Therefore if DM^c uses an LDPC code of rate 1/4, the length of the repetition code only for this scheme has to be shortened to $l = \mathrm{L}/4$.

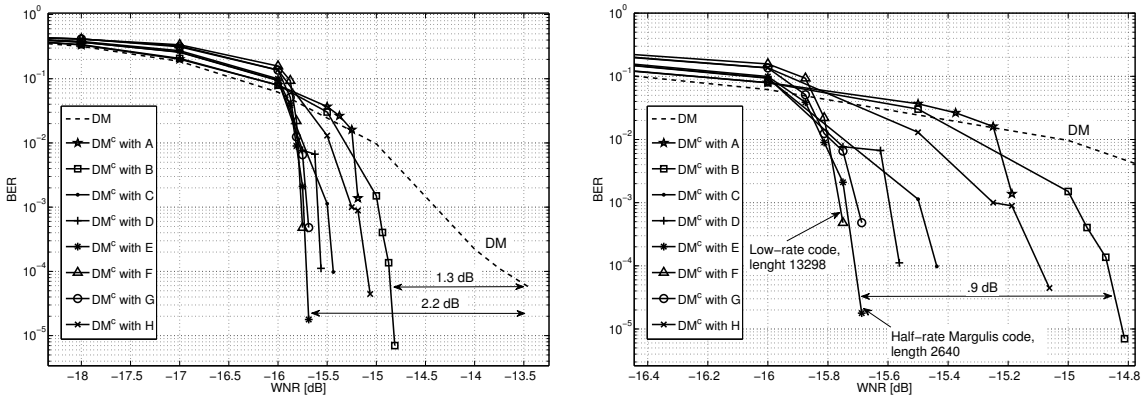
Fig. 3.10 shows a simulation where uncoded DM is compared with DM^c against AWGN. That is, the watermarked audio in time domain is contaminated with Gaussian noise. The step size and the embedding rate were set for all the curves to $\Delta = .02$ and 20 bps respectively. The LDPC codes showed in the simulation were chosen as the best results from a cluster of 30 codes, and they are detailed in Table 3.1. All DM^c algorithms obtained better robustness than uncoded DM (dotted line). The worst DM^c, which uses a high-rate LDPC code, has a benefit of 1.3 dB in comparison with uncoded DM and the best cases reach benefits up to 2.2 dB.

The difference between the worst and the best result of DM^c schemes is around .9 dB. The best results were achieved with codes E and F. The former is a Margulis half-rate code with length 2640 and the latter is a low-rate code of length 13298.

Now, we will add DC property to DM^c. The embedding procedure, for this instance, is the same as Section 3.2.1 and the embedding function is given by (3.2). However the watermark has to be encoded with LDPC codes prior the embedding.

Table 3.1: Characteristics of the LDPC codes used in the simulations.

	Rate	Code length		Rate	Code length
A	.87	3584	E	.5	2640
B	.87	495	F	.24	13298
C	.5	504	G	.25	2000
D	.5	816	H	.26	200



(a) Coded and Uncoded DM.

(b) Closeup of DM with LDPC codes.

Fig. 3.10: Comparison of DM with LDPC codes (DM^c) and uncoded DM over AWGN channel. The details of the LDPC codes are reported in Table 3.1.

The robustness of dither modulation with DC (DCDM) is controlled by the scaling parameter α that must be computed before embedding. The optimal α^*_{opt} depends on the scenario and can be computed with (3.3). As shown in Fig. 3.6 the best performance of DCDM is obtained with α^*_{opt}.

Fig. 3.11 shows the performance of DCDM^c and the uncoded cases. Only the two best LDPC codes from the previous simulation, E and F, were used. DCDM^c increased the robustness up to WNR $= -19.7$ dB which is a benefit of 5.7 dB compared with uncoded DM. An important property is that, due to the LDPC correction capabilities, DCDM^c performs better with a scaling parameter α slightly lower than the optimal $\alpha < \alpha^*_{opt}$. DCDM^c with $\alpha = .5$ is .7 dB better than DCDM^c with α^*_{opt}. The codes E and F perform very similar with α^*_{opt}. There is a minimum advantage, less than .3 dB, of code F with $\alpha = .5$ over code E. Finally, DCDM^c is almost 4 dB better than DM^c.

3.3.2 Spread-Transform dither modulation with LDPC codes

Based on the watermarking scenario of Fig. 3.9, we change the embedding function from DM to spread-transform dither modulation (ST). In this case the coded watermark $\bar{\mathbf{m}}$ is embedded in the projection $\mathbf{s}$ of coefficients $\mathbf{x}$. The embedding follows the same procedure already explained in Section 3.2.2. However the length of the repetition code will be variable

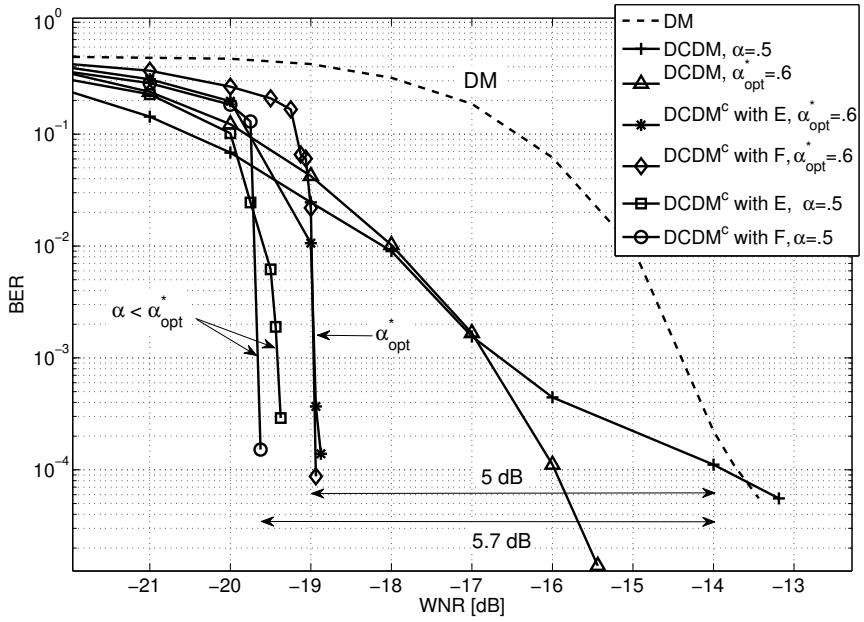


Fig. 3.11: DCDM^c performs better using a scaling parameter slightly lower than the optimal $\alpha < \alpha^*_{opt}$.

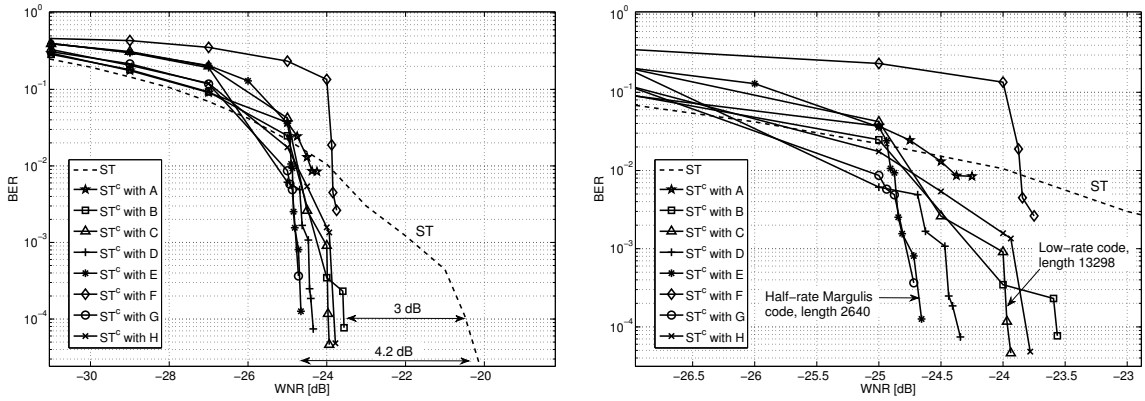
depending on the LDPC code. In that way, we keep the same embedding rate of 20 bps.

At the decoder, the detail coefficients $\mathbf{y}$ are obtained and its projection $\mathbf{s}$ is computed. Using (3.1) and $\mathbf{s}$ the distance $\mathbf{d}$ is calculated, subsequently the **llr** is computed, (3.6), and finally the **LLR** is obtained, (3.7). The embedding-extraction procedures of ST^c are very similar to DM^c, the only difference is the embedding function.

Using the same LDPC codes described in Table 3.1, uncoded ST and ST^c were tested over AWGN channel. Fig. 3.12 shows the comparison. As expected, ST^c performs better than uncoded ST. The best results have a benefit of 4.2 dB and the worst of 3 dB, Fig. 3.12a.

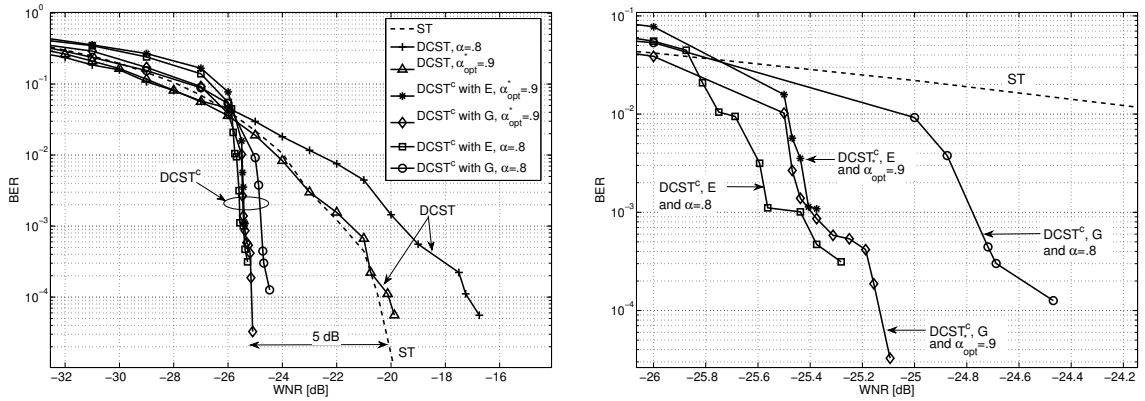
In the previous simulation, Fig. 3.10, the codes E and F obtained the best results. In ST, the code E has again good performance but the result obtained with code F is not as good as before. Other LDPC code with good performance was the code G which is a low-rate code with length 2000.

Applying DC property to ST^c (DCST^c), we obtained the results pictured in Fig. 3.13. The codes used in this simulation were codes E and G which were the best results of ST^c. The best performance of STDC^c, Fig. 3.13b, is obtained with code E, it achieves a robustness of -25.3 dB using an $\alpha = .8$ again lower than the optimal α^*_{opt}. The code G with optimal α^*_{opt} has an acceptable performance but G could not obtain an improvement using a scaling factor lower than the optimal, $\alpha = .8 < \alpha^*_{opt}$.



(a) Coded and uncoded ST. (b) Closeup of ST with LDPC codes.

Fig. 3.12: Performance of ST using LDPC codes (ST^c) and uncoded ST over AWGN channel.



(a) Uncoded and coded DCST (b) Closeup to coded DCST

Fig. 3.13: Performance of coded DCST ($DCST^c$) against AWGN channel.

3.4 General analysis of QIM-based algorithms

The previous Section shows the analysis of dither modulation (DC) and spread-transform dither modulation (ST), both them with improvements of distortion-compensation (DC). We also introduced coded versions of those algorithms using LDPC codes.

In this Section, we perform a comparison between all of them and with BCH codes. Furthermore, we measure the robustness of those schemes against MP3 compression and low-pass filtering. Finally the algorithms are tested with Audio StirMark [54] benchmark.

The best LDPC code from the previous Section was E, see Table 3.1, because its length of 2640 is not too large and it keeps good performance with both DM and ST. Other two codes with good results were F and G, both them are low-rate codes with length 13298 and 2000 respectively. However, they only have good performance with DM or ST but not with both. Besides, the code F has a very large code length that cannot be suitable for watermarking purpose due to constraints of length in the digital work.

In the next comparisons we only include coded cases using the LDPC code E which is the best LDPC code from our experiments in Section 3.3. Fig. 3.14 shows the performance of all

methods introduced in this Chapter and summarized in Table 3.2.

Table 3.2: Summary of the proposals analyzed in Chapter 3 for audio watermarking.

Algorithm	**Description**
DM	Algorithm based on dither modulation.
ST	Algorithm based on spread-transform dither modulation.
DCDM	Dither modulation algorithm improved with distortion-compensation.
DCST	Spread-transform dither modulation improved with distortion compensation.
DM^c	Algorithm based on dither modulation and coded watermark with LDPC codes.
ST^c	Algorithm based on spread-transform dither modulation and coded watermark with LDPC codes.
$DCDM^c$	Coded dither modulation with distortion-compensation property. The coded watermark is obtained with LDPC codes.
$DCST^c$	Coded spread-transform dither modulation with distortion-compensation property. The coded watermark is obtained with LDPC codes.

The first conclusion about Fig. 3.14 is that ST-based performs better than DM-based, the benefit of the best ST-based result over the best DM-based result is 5.9 dB for AWGN channel. The best result from all the methods is for $DCST^c$ with $\alpha = .8 < \alpha^*_{opt}$ that achieves a robustness of WNR $= -25.28$ dB with an error probability lower than 10^{-3}. This represent a benefit of 4.2 dB compared with uncoded ST.

Related with DM, the best method was DCDM with $\alpha = .4 < \alpha^*_{opt}$ which obtained a robustness of WNR $= -19.38$ dB with error probability lower than 10^{-4}. The benefit is of 5.38 dB compared with uncoded DM.

Finally the best result among all methods is $DCST^c$ which performs 11 dB better than uncoded DM.

In Fig. 3.15, we show a comparison of the algorithms using LDPC code E and a half-rate BCH code (1023,503). In general the performance using BCH codes is poor. For example, in DM with BCH codes the improvement over uncoded DM is only of 1.19 dB at 10^{-4}. Using LDPC codes the performance of $DCDM^c$ is improved by 3.32 dB compared with DCDM with BCH codes. Results related with ST confirm the better performance of LDPC codes. $DCST^c$ has a benefit of 2.53 dB compared with the same method with BCH codes. Finally, due to the lower error correction capability of BCH codes, DCST with $\alpha = .8 < \alpha^*_{opt}$ is not capable achieve better performance than ST.

ST-based algorithms have better performance than DM-based algorithms for AWGN channels. This advantage is increased when the SNR of the watermarked files was measured. In DM-based algorithms the average SNR obtained was 33 dB and with ST-based the SNR achieved was, in average, 51 dB. Thus, ST-based algorithm proved to be more robust than DM-based and furthermore, they produce less distortion in the audio signal.

Until now the analysis has been based on AWGN channels, however the audio files are

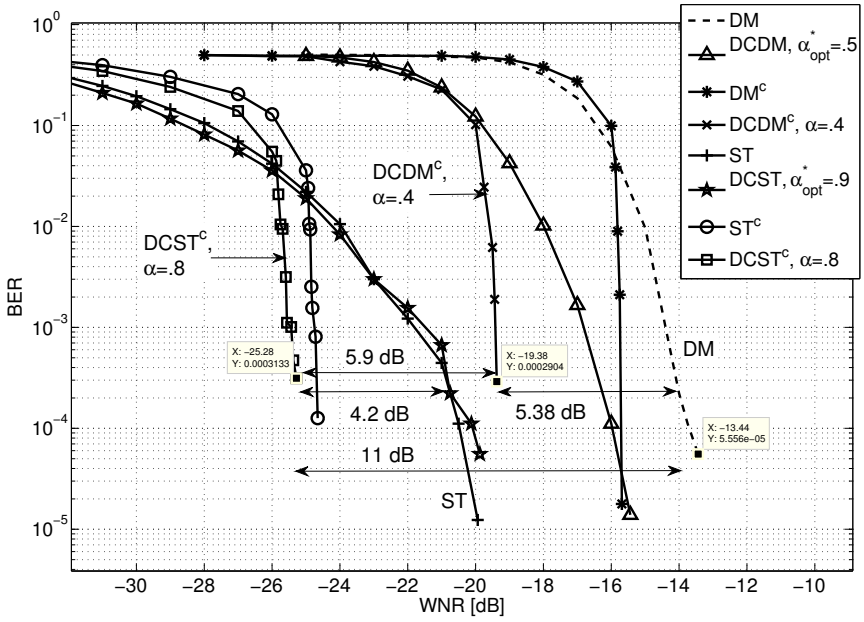


Fig. 3.14: Comparison of the best results of uncoded algorithms and coded algorithms using the LDPC code E from Table 3.1.

susceptible to different distortions produced by common signal processing. Therefore, in order to complete our analysis Fig. 3.16 shows the robustness of the algorithms against MP3 compression and low-pass filtering. In contrast with the results obtained for AWGN where ST-based algorithms were better, DM-based algorithms are more robust against common signal processing.

DCDM^c is the algorithm with the best result after MP3 compression, the watermark resist compressions up to 80 kbps with very low BER ≈ 0 and for the lowest compression with commercial value, 64 kbps, the BER $= .1$ is still not high for watermarking standards, Fig. 3.16a. When low-pass filtering is applied to the algorithms, DCDM^c and DCST^c have very similar performance but DCDM^c is still slightly better, Fig. 3.16b. The watermark can be recovered with high reliability from a frequency cutoff of 1.5 kHz and also it supports cutoff of 800 Hz with BER $\approx .1$.

Finally the watermarked audio was tested with Audio StirMark benchmark. StirMark applies different attacks to the audio file with the purpose to erase the watermark from the audio. The attacks are varied, they ranging from attacks in time, frequency, attacks that add noise or more complex like audio denoising, and even attacks that desynchronize the detection of the watermark.

Table 3.3 summarizes the results of StirMark benchmark for two algorithms DCDM^c and DCST^c, both them with LDPC code E which obtained the best results from the previous analysis. DM-based algorithms again show more robustness than ST-based algorithms for common signal processing. Attacks like *copysample*, *cutsamples* and *zero-attacks* removed

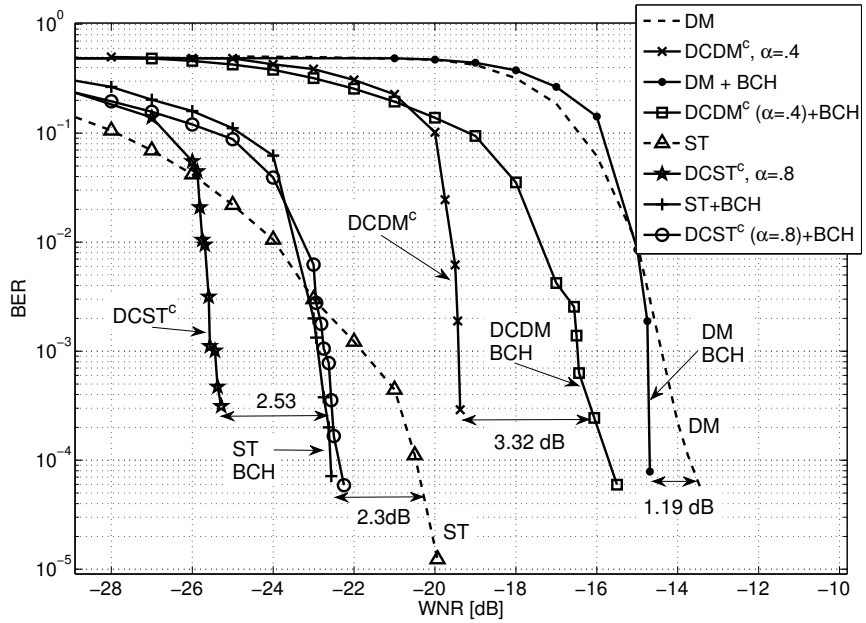
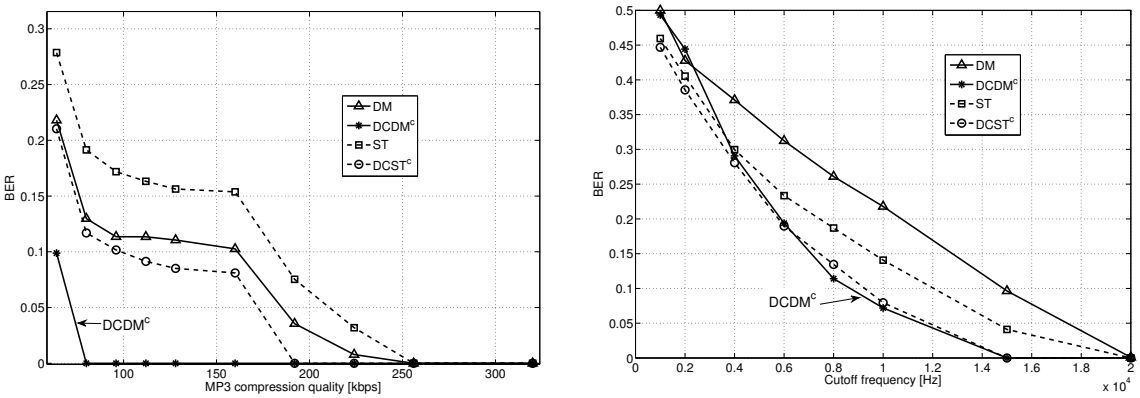


Fig. 3.15: Comparison of our proposals using LDPC code E and BCH codes.



(a) MP3 compression.

(b) Low-pass filtering.

Fig. 3.16: Robustness against MP3 compression and low-pass filtering.

totally the watermark from the audio file because they produce desynchronization and, in this Chapter, we assumed that both decoder and encoder were in synchrony. Therefore without sync control, the desynchronization attacks tend to have very high BER.

In the beginning of the analysis, ST-based algorithms showed good performance against AWGN. However when more common attacks were applied to the audio files, the robustness of ST-based algorithms decreased dramatically. On the other hand, DM-based algorithm had acceptable robustness against AWGN and good robustness with common attacks, especially against MP3 compression which is a very popular compression algorithm for audio. Moreover, DM algorithms have less computational complexity than ST algorithms.

Table 3.3: Results of StirMark Audio benchmark for the best methods using dither modulation (DM) and spread-transform (ST). Details and description of each attack can be found in [54] and [55].

Attack	**DCDM^c**, $\alpha = .5$ [BER]	**DCST^c**, $\alpha = .9$ [BER]	**Attack**	**DCDM^c**, $\alpha = .5$ [BER]	**DCST^c**, $\alpha = .9$ [BER]
addbrumm 100	0.0000	0.5288	addbrumm 4100	0.0000	0.5076
addbrumm 9100	0.3321	0.4727	addbrumm 10100	0.2773	0.5048
addnoise 100	0.0000	0.0000	addnoise 900	0.0000	0.4391
addsinus	0.3328	0.3912	amplify	0.1970	0.4455
compressor	0.0000	0.0000	copysample	0.5018	0.5013
cutsamples	0.4960	0.4952	dynnoise	0.0000	0.4879
echo	0.5449	0.4894	exchange	0.0000	0.0000
extrastereo 30	0.0000	0.0000	extrastereo 50	0.0000	0.0000
extrastereo 70	0.0000	0.0000	fft hlpass	0.1045	0.5265
fft invert	0.4253	0.0000	fft real reverse	0.0000	0.1053
fft stat1	0.3477	0.4025	fft test	0.3475	0.3952
flippsample	0.0604	0.2949	invert	0.4177	0.0000
lsbzero	0.0000	0.0000	normalize	0.4513	0.6139
nothing	0.0000	0.0000	original	0.0000	0.0000
rc highpass	0.2788	0.5098	rc lowpass	0.0000	0.0000
resampling	0.0000	0.0000	smooth2	0.0000	0.1149
smooth	0.0000	0.0000	stat2	0.0000	0.0000
stat1	0.0000	0.0000	voiceremove	0.4937	0.5114
zerocross	0.5293	0.5088	zerolength	0.5088	0.5035
zeroremove	0.5035	0.4962			

3.5 Further improvements

Traditional dither modulation (DM) uses uniform quantization step size Δ. However, since the audio is not a stationary signal there are audio segments where the watermark can be embedded stronger than others. Thus, in this Section we propose an adaptive dither modulation (ADM) which uses a variable quantization size δ.

Similar proposal are described in [56] and [57]. In the former, Li and Cox presented an adaptive algorithm based on QIM for still images. The adaptive step size was computed with Watson's perceptual model which is inherent to images and cannot be used on audio. The results show an improvement in the image quality, achieving the same or better robustness than traditional QIM. The latter is a proposal for audio introduced by Li and Yu. They used the psychoacoustic model generated by MPEG-1 to control the watermark strength in spread spectrum watermarking.

Our algorithm embeds the watermark in frequency domain using ADM. The adaptation of the step size δ depends on the masking threshold computed with the aid of the analysis of MPEG-1 layer I, and the frequency coefficients were computed with the discrete wavelet

packet transform (DWPT). LDPC codes were used to encode the watermark, however two different kernels to compute the log-likelihood ratio, **llr**, were implemented. Fig. 3.17 is a block diagram with the most important steps of our proposal.

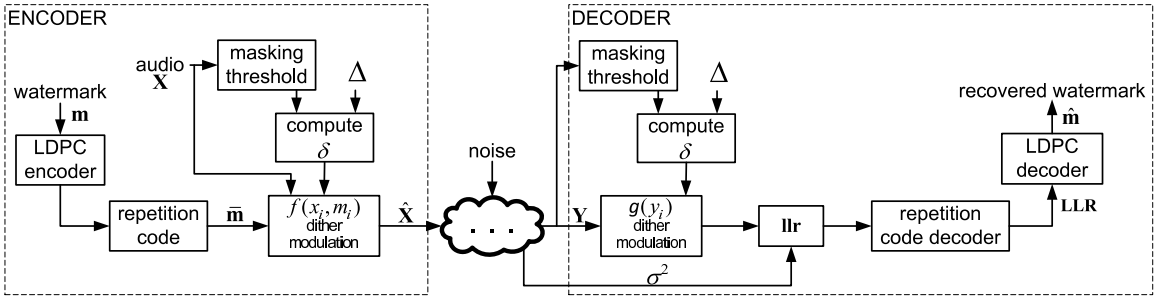


Fig. 3.17: General block diagram of adaptive dither modulation.

DWT produces a decomposition oriented only to low frequencies. DWPT offers richer frequency decomposition because unlike DWT, DWPT also decompose the high frequencies of a signal. If we picture the decomposition like a binary tree, DWPT produces 2^k sub-bands were k is the desired level of decomposition. Fig. 3.18 represents the decomposition tree of a segment of audio signal until the fifth level using DWPT. As result, the audio is divided into $2^5 = 32$ sub-bands, each of them containing certain range of frequencies.

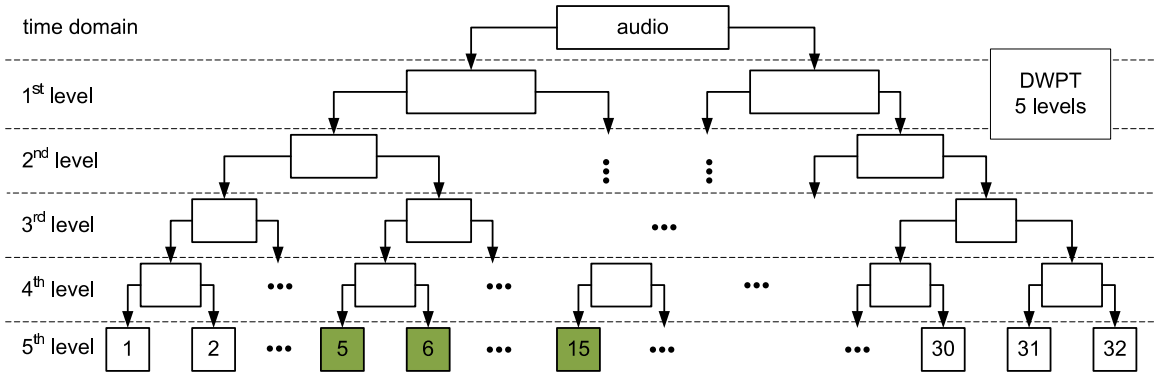


Fig. 3.18: Discrete wavelet packet transform of an audio signal until the 5$^{\text{th}}$ level.

Before the embedding, the watermark $\mathbf{m}$ is encoded with a concatenation of LDPC code and repetition code, in the same fashion as Section 3.3.1, producing a coded watermark $\bar{\mathbf{m}}$. According to the analysis of the aforementioned Section, the best code for DM was a half-rate Margulis code with length of 2640, code E of Table 3.1. Thus, in this algorithm only the code E is used.

The embedding phase starts dividing the audio into non-overlapped blocks T_i of 512 samples. Each block T_i is forwarded in parallel to the wavelet decomposition and to the masking analysis. The block T_i is decomposed with *Haar* mother wavelet and DWPT until the fifth level producing 32 sub-bands with 16 coefficients each. Parallel to the wavelet analysis, the same block T_i is analyzed with the psychoacoustic model of MPEG-1 layer I

to compute the minimum masking threshold of that block. Based on the minimum masking threshold of the block T_i and following the procedure described in [57], the parameter a_j is calculated for each wavelet sub-band of the block T_i, where $j \in [1, 32]$. Thus, a different quantization step size δ_j^i is computed for every wavelet sub-band j of the i-th block, with $\delta_j^i = a_j * \Delta$.

For this proposal, Δ is a global constant known in both encoder/decoder, and is analogous to the step size Δ of traditional DM which also has to be known during the whole watermarking process. The constant Δ is empirically adjusted to meet the requirements of imperceptibility of the watermark. In our experiments Δ was set to be .025 producing the audio quality of SNR = 27 dB.

The watermark $\bar{\mathbf{m}}$ is embedded in the coefficients $\mathbf{x} = x_1, x_2, \ldots$ of the wavelet sub-bands 5 to 15 as shown in Fig. 3.18. The embedding function is DM with the adaptive step size δ:

$$\hat{x}_i = Q\big(x_i + v(\bar{m}_i), \delta_j\big) - v(\bar{m}_i),$$

where $\hat{x}_i$ is the watermarked coefficient and $v(\cdot)$ is a dither parameter dependent of $\bar{m}_i$ and explained in Section 2.3.1. Then, $\mathbf{x}$ is replaced with the watermarked coefficients $\hat{\mathbf{x}}$ and the audio is reconstructed with the inverse DWPT.

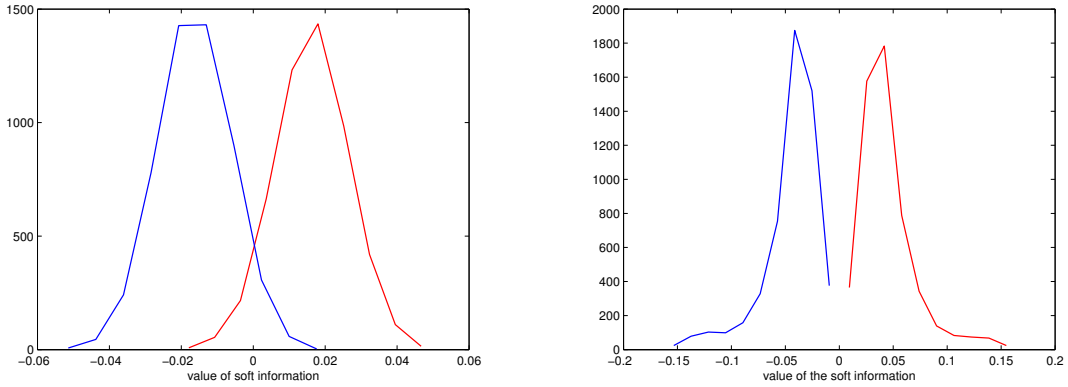
The decoder assumes an audio file $\mathbf{Y}$ that might have degradations due to attacks or noise. The audio is divided into blocks, the wavelet coefficients are computed and also the adaptive step size δ has to be obtained from the minimum masking threshold. The decoding process is very similar to the embedding phase.

Once the frequency coefficients $\mathbf{y}$ and the variable step size δ were already obtained, the distance between $\mathbf{y}$ and its quantization respect to the reconstruction points belonging to the watermark symbol 0 is computed

$$d_i = |y_i - Q(y_i + v(0), \delta_j) + v(0)|, \tag{3.8}$$

and then, the soft-information is $\mathbf{r} = \mathbf{d} - \delta_j/4$.

Using $\mathbf{r}$, we need to compute the log-likelihood ratio $\mathbf{llr}$ which is used by the SPA of the LDPC decoder to recover the watermark $\hat{\mathbf{m}}$. In traditional DM, the $\mathbf{llr}$ can be fairly computed with a Gaussian kernel because the probability density function (PDF) of the soft-information $\mathbf{r}$ after AWGN addition behaves like Gaussian variable. This behavior can be seen in Fig. 3.19a. However when ADM is used the PDF of $\mathbf{r}$ is more similar to a Rayleigh fading distribution, see Fig. 3.19b, even if the audio is contaminated with AWGN in time domain. Thus, using a Rayleigh fading kernel we expect to compute more accurate "a priori" probabilities of the received bits.



(a) PDF of dither modulation (DM).

(b) PDF of adaptive DM (ADM).

Fig. 3.19: Probability density function of the soft-information extracted with DM.

Computing the **llr** with a Rayleigh fading kernel, we have:

$$llr_i = \frac{2}{\sigma^2} r_i a, \tag{3.9}$$

where σ^2 is the noise variance computed with

$$\sigma^2 = \frac{4-\pi}{2} a^2,$$

and a is a parameter estimated by

$$a = \sqrt{\frac{1}{2N} \sum_{i=1}^{N} r_i^2} \qquad \text{for } N = |r|.$$

Finally, the repetition code is decoded with (3.7), the **LLR** is obtained and forwarded to the LDPC decoder. The embedding rate achieved was of 43 bps that is 23 bps better than the schemes previously explained in this Chapter.

The first simulation, Fig. 3.20, shows a comparison of uncoded DM and uncoded ADM for AWGN. Uncoded means that LDPC codes were not used but due to the weakness of DM, repetition codes had to be used. ADM and DM produced the same audio quality, SNR = 27 dB, however ADM is more robust than DM. ADM has much lower BER with high noise variance, however when the noise is very small the performance of ADM is decreased. This effect is because there are audio segments with bad masking properties and therefore ADM uses small step size δ to keep the imperceptibility of the watermark.

Fig. 3.21 shows the histogram of the step size δ used in ADM. The good robustness of ADM is because we are able to use big quantization step $\delta > .2$ in some parts of the audio file. However there are places when δ has to be very small and therefore even in situations when the noise is small, we still find a few errors. The solution is to use an error control code

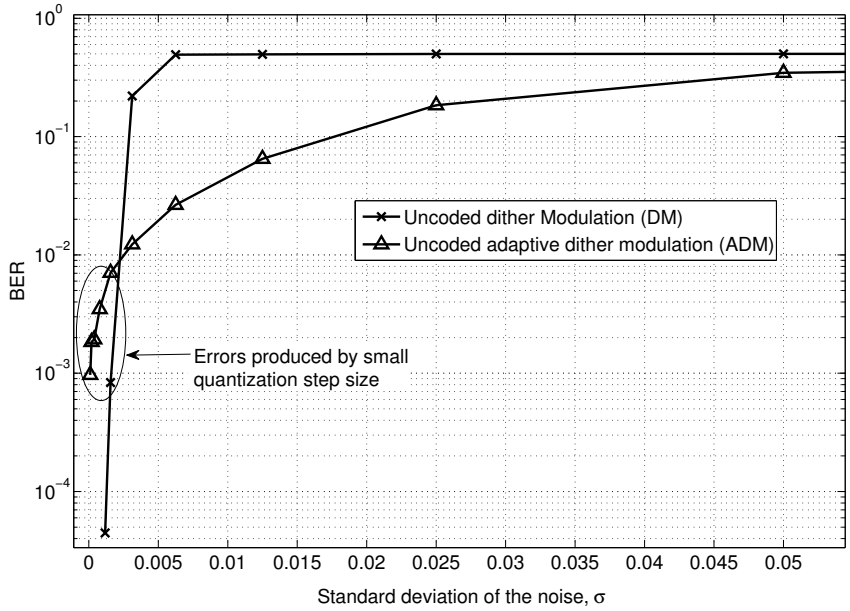


Fig. 3.20: Performance of traditional DM and ADM for AWGN.

like LDPC codes to correct those errors.

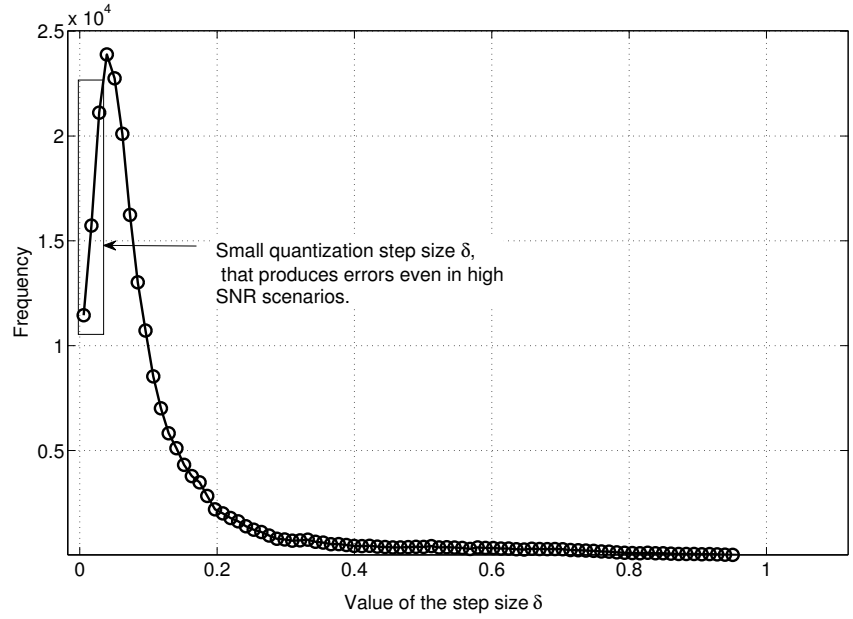


Fig. 3.21: Histogram of the step size δ used in ADM.

The last simulation on Fig. 3.22 shows the performance of ADM, over AWGN channel, using different kernels to compute the probabilities of the received symbols.

First, ADM was decoded using a Gaussian kernel calculated with (3.6). The addition of LDPC codes correct the errors produced by smalls δ and clearly increase the robustness up to $\sigma^2 = 0.005$. In order to obtain fair comparison, the payload of both methods, uncoded and coded ADM, is the same.

Second, ADM was decoded with a Rayleigh fading kernel increasing even more the robustness of the watermark. Thus, the noise in ADM is better modeled using a Rayleigh fading channel. The watermark achieved robustness up to $\sigma^2 = 0.01$ of AWGN with a BER lower than 10^{-3}.

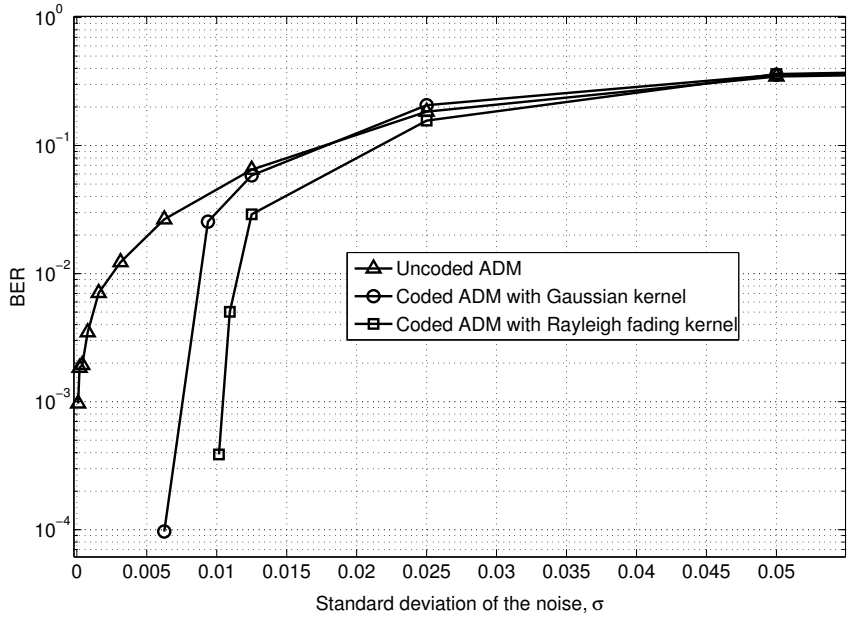


Fig. 3.22: Coded adaptive dither modulation using Gaussian kernel and Rayleigh fading kernel.

3.6 Conclusion

We have defined a framework for digital audio watermarking. The framework has an embedding in frequency domain, QIM-based algorithms as embedding functions and coded watermarks produced by concatenation of LDPC codes and repetition codes.

The aforementioned framework has been extensively analyzed. The decomposition of the audio file in its frequency components was studied with the DWT and DWPT. Several QIM-based algorithms like dither modulation, spread transform and its improvements with distortion-compensation were implemented. The coded watermarks were produced with many combinations of LDPC codes and repetition codes of which, a half-rate Margulis code with length 2640 was the best code.

From the analysis, ST-based algorithms show better performance than DM-based algorithms when the audio file was contaminated with AWGN in time domain. However when the watermarked audio was tested with common audio signal processing like compression and filtering, DM-based algorithms were more robust especially against MP3 compression, achieving a compression quality of 80 Kbps with BER ≈ 0. The better robustness of DM-based over ST-based algorithms was confirmed with the StirMark benchmark, where DM obtained

again better results than ST.

Finally, we proposed further improvements using an adaptive version of DM. This algorithm produces better audio quality than traditional DM with the same or better robustness. The adaption was controlled with the minimum masking threshold computed with MPEG-1 layer I and the decoding was approached with a Rayleigh fading kernel which produced better results than using a Gaussian one. Since the algorithm used DWPT, the audio was divided into more frequencies that allowed us to embed more information. Thus the payload of this algorithm achieved 43 bps.

4

High payload watermarks for MP3 compression

MPEG-1 layer III (MP3) is a popular format of compressed digital audio because offers good audio quality within small storage space. These characteristics have been used by on-line stores to sell and deliver audio files in a short period of time through the Internet. However those same characteristics, make easier the unauthorized copying of audio files. Moreover, MP3 compression is considered a tough attack to audio watermarking.

Our aim in this Chapter is to develop audio watermark algorithms robust to compression with the lowest commercial value (64 kbps) and with high embedding payload. This scenario is pictured in Fig. 4.1. Security-related applications of watermarking usually require a few bits but if the embedding capacity[1] is increased, then more applications can be benefited from watermarking.

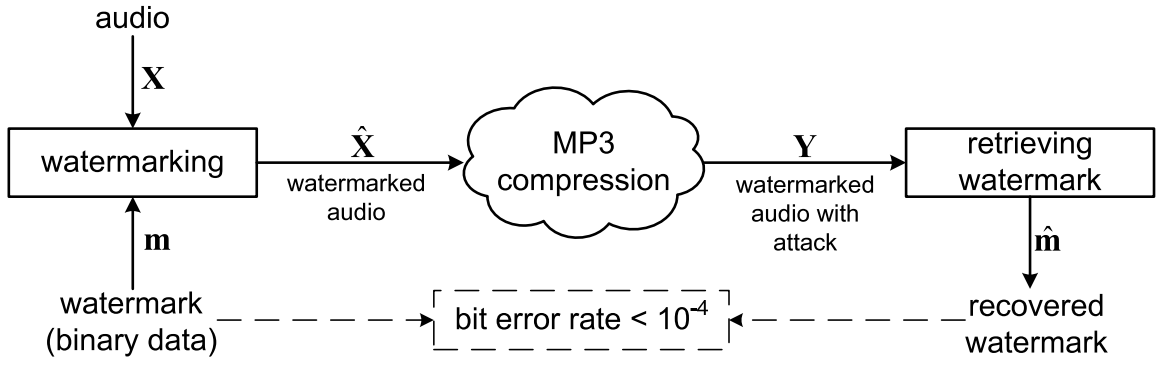


Fig. 4.1: General diagram of the studied system.

Since the pioneer paper of Boney [10] in 1996 for audio watermarking, many algorithms with robustness to MP3 compression have been proposed. For example in 2003, Cvejic [31]

[1]The term "capacity" refers to watermark payload and it is different from the theoretical channel capacity defined by information theory.

proposed a scheme using spread spectrum and characterization of the attack, with a payload of 27.1 bps. In the same year, In-Kwon [58] used patchwork algorithm to embed the watermark in audio files producing a scheme with 10 bps. Later, in 2004, Wang [59] introduced a decoding algorithm using linear predictive coding to recover the watermark from wavelet domain; the achieved payload was 10.72 bps. Another two proposals [60] and [61] were described in 2006, the former includes neural networks to increase the robustness and the latter proposes a solution for time-scale modification, obtaining 86 bps and 4.26 bps respectively. Between 2007 and 2008, Xiang presented two schemes in [62] and [63] based on modifying the histogram of the audio files; with embedding rates of 3 bps and 2 bps. At the same time, two papers with high embedding rates were published, [64] with a payload of 170 bps and [40] with 220 bps; however those payloads were achieved with MP3 of 128 kbps and 96 kbps respectively. More recently, in 2009 Fan [65] proposed an embedding of a chaotic-based watermark on discrete fractional sine transform domain, with 86 bps of payload. Also, in the same year, Wang [66] introduced a robust algorithm against MP3 at 64 kbps but the embedding payload was not reported. Finally, in 2010 a self-synchronized algorithm was introduced by Megías [67] with an embedding rate of 30.09 bps. Almost all the works aforementioned have an embedding rate lower than 100 bps and those with payload higher than 100 bps are not robust to MP3 with quality of 64 kbps.

We use dither modulation (DM) to embed a coded-watermark in wavelet domain. Similar proposals can be found in [68] and [41]. The former is an algorithm with self-synchronization in wavelet domain that embeds the watermark in the low-frequency coefficients. The scheme achieves 172 bps, however is well-known that modifications to low-frequencies produce audible distortion, and the authors did not report evidence of the audio quality. The latter is a recent proposal, 2010, that embeds the watermark by quantizing the Euclidean norm of a singular value decomposition of an audio segment. The reported payload is 196 kbps, however the bit error probabilities are computed with short simulations, around 5 seconds.

Our algorithm embeds the coded-watermark in a specific range of frequencies which were found to be robust against MP3, Section 4.1. The coded-watermark was obtained with the concatenation of an outer LDPC encoder and inner repetition codes. Watermarking channels tend to have very high BER, therefore repetition codes are needed to increase the SNR [28]. Audio watermarking with powerful ECC can be found in [31] where Turbo codes together with spread spectrum are used achieving 21.6 bps. LDPC codes have been already used, but in image watermarking [32] where they produced very high embedding rates. Nevertheless only normal decoding was applied.

Our main algorithm is explained in Section 4.2. The decoder is implemented in two versions: semi-blind and blind decoding. In semi-blind decoding, the "watermarked audio without noise" is compared with the noisy audio in order to compute the noise variance. The

statistics of the channel noise, variance, can be approximated with different techniques that are not the aim of this Chapter. However we implemented a blind decoding with the aid of pilot symbols.

The proposed algorithm achieves an embedding capacity of 155 bps with blind decoding and 204 with semi-blind decoding, shown in Section 4.3. Then, the proposal was improved using erasures at LDPC decoder, Section 4.4. The result is an embedding capacity of 229.7 bps which represents a benefit of more than 15 bps in comparison with normal decoding and more than 33 bps compared with any other proposal robust to MP3 at 64 kbps, Section 4.5.

It is worth to mention that there are very high payload techniques for data hiding in audio which achieve embedding rates of 689 bps in [69], 2996 bps in [70] and 11000 bps in [71] but they are not robust to MP3 of 64 kbps.

Finally, the last two Sections, 4.6 and 4.7, are dedicated to the audio quality and conclusions respectively.

4.1 Repercussions of MP3 on wavelet domain

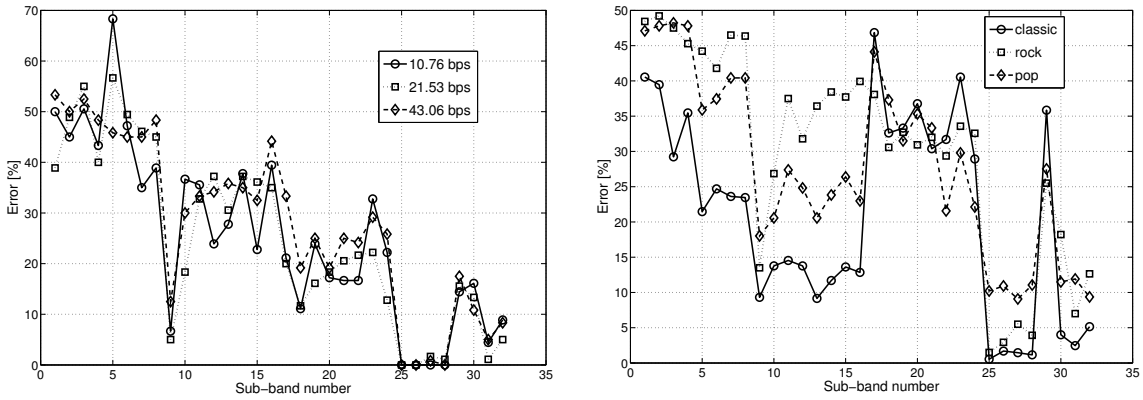
We are focused in one of the most popular manipulation on audio files, MP3 compression. The key to MP3 is lossy. Nonetheless, this algorithm can give transparent, perceptually lossless compression. To achieve this transparency, the compression is done according to the human auditory system which has different responses depending on the frequency. Therefore, it is expected that different range of frequencies of the audio files will be affected with different intensity.

When the audio file is decomposed with wavelet packet, each sub-band represents a different range of frequencies. We are interested in measuring the distortion of different sub-bands after MP3 compression to find suitable sub-bands for embedding.

The audio files were divided in blocks of 512 samples and wavelet packet decomposition was applied to each block using 5 levels. Simple *Haar* was used as mother wavelet. After decomposition, 32 wavelet sub-bands are obtained. The binary watermark was embedded in each sub-band individually using DM and repetition codes, no other ECC was involved. Finally, the audio files were reconstructed and attacked with MP3 compression at 128 kbps.

Two experiments were conducted. First, Fig. 4.2a shows the error percentage of each wavelet sub-band for three different embedding rates 10.76, 21.53 and 43.06 bps. Classic music: *Egmont Op. 84* mono, 44.1 kHz/16 bits and with 8 minutes long was used. Second, Fig. 4.2b shows the bit error percentage of each wavelet sub-band for three different music genres with an embedding rate of 21.53 bps. Every music gender was a cluster with 10 different audio files of 2 minutes long each.

In both results of Fig. 4.2, the wavelet sub-bands 25, 26, 27 and 28 obtained better



(a) Using different embedding rates.

(b) With different music.

Fig. 4.2: Error percentage of the wavelet sub-bands after MP3 at 128 kbps.

robustness against compression, those sub-bands belong to frequencies ranging from 11 kHz to 13.7 kHz.

With Fig. 4.2a is guaranteed that the sub-band robustness behaves similar for different embedding rates, and then, the result is expanded to different music with Fig. 4.2b. In the simulations, Δ was fixed according to SNR > 30 dB. We define SNR as the ratio between original audio signal power and that of the audio file with information embedded, see (2.9).

4.2 High payload audio watermark algorithm

The main idea is to embed a coded-watermark inside wavelet domain using DM. The coded-watermark is obtained by concatenation of an outer LDPC code and inner repetition codes. The watermark is decoded by retrieving soft-information with DM, computing the metric, and finally decoding the metric with sum-product [13] algorithm. The statistics of the channel noise, noise variance σ^2, are involved in the computation of the metric and the performance of the sum-product algorithm depends highly in a good estimation of σ^2. These ideas are summarized as block diagram in Fig. 4.3.

Two different decoding strategies are introduced. The first one considers general statistics of the channel noise to compute the metric, i.e. the log-likelihood ratio **llr**, and the second one computes independent statistics about the noise for different segments along the audio file.

4.2.1 Embedding

In the embedding process, the audio file in time domain $\mathbf{X}$ is divided in non-overlapped blocks, T_h, of 512 samples. Wavelet packet decomposition is applied in five levels using simple *Haar* mother wavelet. From each time-domain block T_h, 32 wavelet sub-bands with 16 coefficients each are obtained. According to Section 4.1, only coefficients from the sub-

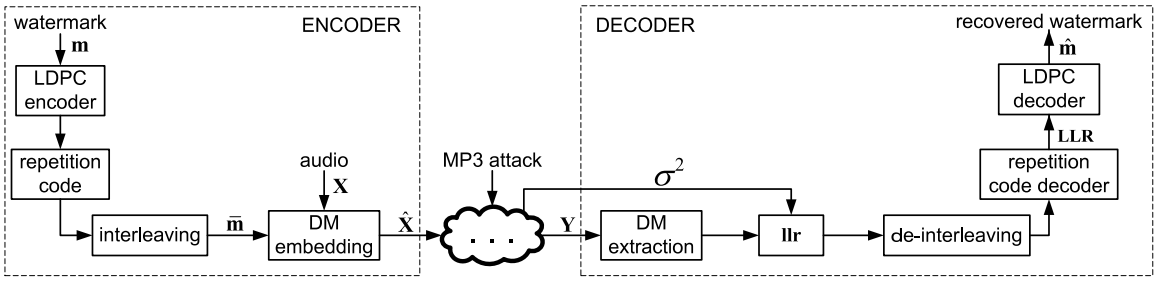


Fig. 4.3: General watermarking scheme.

bands 25 to 28 will be used for embedding. From now on, we will use the notation "25-28" to means wavelet sub-bands from 25 to 28. Frequency coefficients from 25-28 are arranged in a vector $\mathbf{x} = x_1, x_2, \ldots, x_n$ which contains not only the coefficients from the block T_h but from all blocks ordered according to time.

The binary watermark $\mathbf{m} = m_1, m_2, \ldots$ is firstly encoded with a Margulis half-rate LDPC code with code length of 2640. The output of the LDPC encoder is again encoded with an inner repetition code of length l, the output is permuted with an interleaving and finally the coded-watermark $\bar{\mathbf{m}} = \bar{m}_1, \bar{m}_2, \ldots, \bar{m}_n$ is obtained.

The watermark robustness is variable depending solely on the repetition code because the LDPC code will be the same and also the quantization step size Δ is fixed. If the repetition code is large then the robustness is better but the embedding capacity is lower and vice versa.

Each bit from the coded-watermark $\bar{\mathbf{m}}$ is embedded in one coefficient x_i using DM, $\hat{x}_i = Q\big(x_i + v(\bar{m}_i), \Delta\big) - v(\bar{m}_i)$ where Q represents the quantization function with step Δ, $v(\bar{m}_i)$ is a dithered parameter that modulates the watermark symbol $\bar{m}_i$, and $\hat{x}_i$ is the watermarked coefficient. Then $\mathbf{x}$ is replaced with $\hat{\mathbf{x}}$ and the inverse wavelet packet transform is applied.

The watermarked audio $\hat{\mathbf{X}}$ is susceptible to suffer any sort of degradation due to common signal manipulations, e.g. MP3 compression, or direct attacks which attempt to destroy the watermark. Therefore, $\mathbf{Y}$ is the watermarked audio with degradations or attacks.

4.2.2 Decoding

$\mathbf{Y}$ is divided in blocks $\hat{T}_h$ of 512 samples and wavelet packet in five levels is computed. Watermarked coefficients from 25-28 are extracted and arranged as the vector $\mathbf{y} = y_1, y_2, \ldots, y_n$. The distance $\mathbf{d} = d_1, d_2, \ldots, d_n$ is computed as the absolute value of $\mathbf{y}$ and its closest reconstruction point with respect to $v(0)$, $d_i = \big|y_i - Q\big(y_i + v(0), \Delta\big) + v(0)\big|$.

The soft information $\mathbf{r} = r_1, r_2, \ldots, r_n$ is computed with $\mathbf{r} = \mathbf{d} - \Delta/4$. Heuristically, we have seen that the noise, due to MP3, in the coefficients of 25-28 behave very similar to a Gaussian distribution. Therefore, the **llr** is computed with a Gaussian kernel, $llr_i = \Delta r_i / 2\sigma^2$, where σ^2 is the noise variance.

In the next Subsections, two different decoding variations are explained. However all the

steps explained above are common for them both.

Global Noise Variance

A good estimation of the noise variance σ^2 is important to obtain good performance. In traditional communications many sources of noise can be modeled as Gaussian, moreover in many of them a single noise variance characterizes the channel. Digital watermarking has been shown to be a sort of communications and also the noise in wavelet coefficients, due to MP3, behave similar to Gaussian. Therefore, this approach is an analogy to those communication systems which use a single noise variance to model the noise in the whole channel.

Log-likelihood ratio is computed straightforward with $\mathbf{llr} = \Delta\mathbf{r}/2\sigma^2$. De-interleaving is applied to $\mathbf{llr}$ and the repetition code, of length l, is decoded with $LLR_k = \sum_{i=l(k-1)+1}^{lk} llr_i$. Finally, $\mathbf{LLR} = LLR_1, LLR_2, \ldots$ is forwarded to the LDPC decoder and the watermark $\hat{\mathbf{m}}$ is recovered.

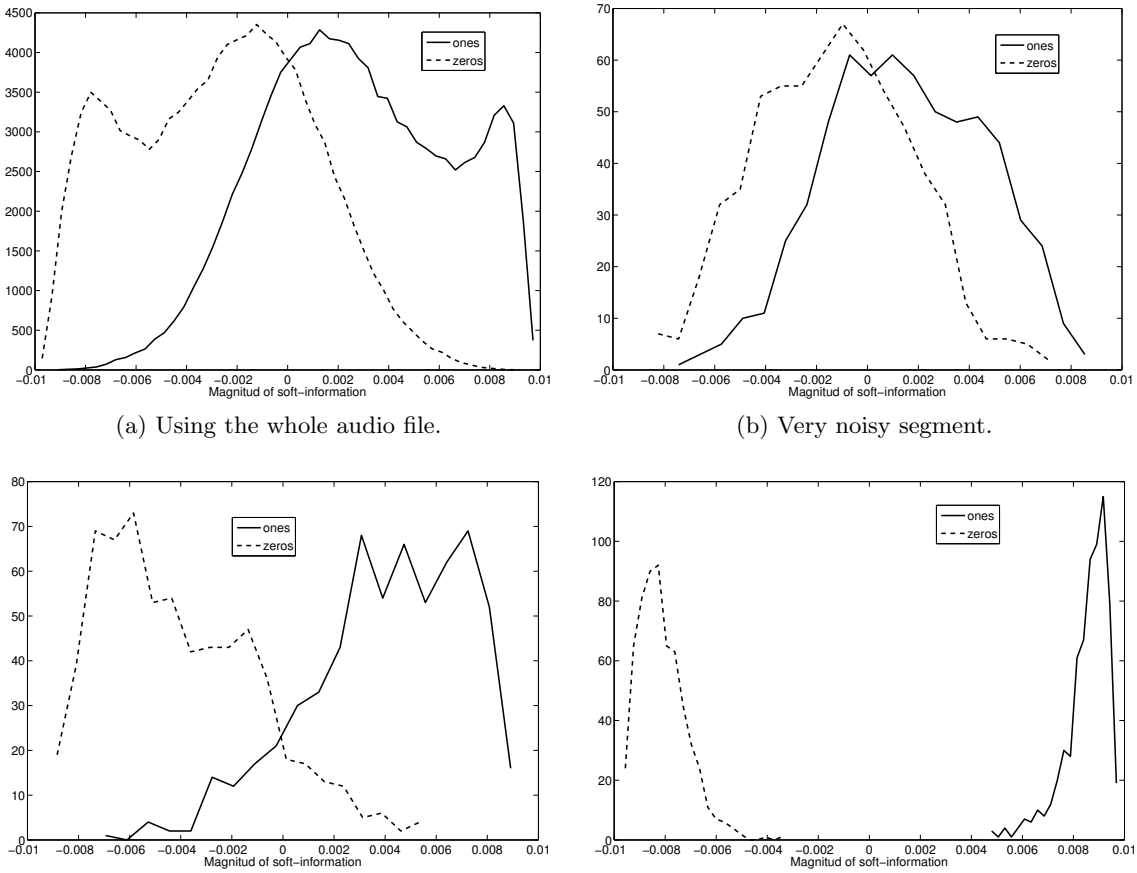
Particular Noise Variance

MP3 takes into account the human auditory system, then the compression is not constant because the audio is not a stationary signal. Therefore, different amount of distortion is expected in different segments of the audio along time.

Fig. 4.4 shows the histograms of the soft-information $\mathbf{r}$ for different segments along the audio after MP3 at 128 kbps. In those graphs, we can distinguish between soft-information which belongs to the embedded bits $\bar{m}_i = 1$ with solid lines and $\bar{m}_i = 0$ with dotted lines. Fig. 4.4a represents the density when all the frequency coefficients from a certain audio are taken into account. That is, the coefficients of 25-28 from all the blocks $\hat{T}_h$ are used. Figs. 4.4b, 4.4c and 4.4d, are densities from small continuous segments of a certain audio file. These segments contain only 320 frequency coefficients, that is, the equivalent to 5 continuous blocks $\hat{T}_h$. The segments were chosen randomly along the audio file.

The audio used to generate Fig. 4.4 was *A change of season* by Dream Theater sampled at 44.1 kHz with 16 bits. The binary information was embedded in sub-bands 25-28 using DM with only repetition codes of length 4, Δ was set to .01.

Figs. 4.4b, 4.4c and 4.4d show that the noise in the audio file due to MP3 is not constant. For example in Fig. 4.4d, the difference between ones and zeros is perfectly distinguishable and therefore, a decoding without errors is expected. Other regions like Fig. 4.4b suffered more distortion and probably many errors will be produced. Nevertheless, if the statistics of the channel noise are computed using the whole audio file, Fig. 4.4a, the reliability is not good.



(a) Using the whole audio file.

(b) Very noisy segment.

(c) Noisy segment.

(d) Noiseless segment.

Fig. 4.4: Histograms of the soft-information obtained from the frequency coefficients after MP3 compression. "a" was computed with the whole audio file. "b", "c" and "d" are histograms from random segments of 320 coefficients.

Based on the previous experiment, will be more reliable to compute individual noise variances for different segments along the audio file. The decoding proposal in this Section aims to compute independent noise variances along time to obtain a better model of the noise produced by MP3.

The log-likelihood ratio **llr** is computed with:

$$llr_i = \frac{\Delta r_i}{2\varsigma^{\mathbf{2}}}, \tag{4.1}$$

where $\varsigma^{\mathbf{2}} = \varsigma_1^2, \varsigma_2^2, \ldots \varsigma_p^2$ is estimated by dividing the soft-information $\mathbf{r}$ in p blocks and computing an individual noise variance for each block. For example, if r_i belongs to the block j therefore $llr_i = (\Delta r_i)/(2\varsigma_j^2)$. Then, de-interleaving is applied to **llr** and the repetition code is decoded with (3.7). The **LLR** is forwarded to the LDPC decoder and the watermark $\hat{\mathbf{m}}$ is recovered.

4.3 Embedding capacity tolerant to MP3

This Section is divided in two parts: decoding for semi-blind schemes and decoding for blind schemes. This division is because an accurate estimation of the noise variance is needed at the decoding part, therefore the difference between them is about how to compute the noise variance. With the semi-blind scheme the decoder has knowledge of the watermarked audio $\hat{\mathbf{X}}$ and it is capable to compute the real noise generated in the audio due to MP3. Semi-blind decoding was developed to show the potential of this proposal.

In blind decoding, an estimation of the noise is needed. Several techniques about channel estimation have been proposed and implemented in practical communications schemes, e.g. "training symbols", "least-squares" or even more complex techniques which involves "turbo equalization".

Our aim in this Chapter is to develop reliable watermark techniques rather that focus on describing the noise generated by MP3 compression. Therefore, we propose a blind detection using a basic technique with *pilot symbols*, nevertheless our blind decoding produces better performance than, in our knowledge, most of the previous watermark schemes for MP3 at 64 kbps.

The audio files used in the simulations have the next characteristics: WAVE files, mono and sampled at 44.1 kHz with 16 bits. All the results are an average of simulations based on three audio files, classic: *Egmont Op. 84* (7 minutes), pop: *Billie Jean* by M. Jackson (4 minutes) and rock: *A change of seasons* by Dream Theater (8 minutes). Those audio pieces were chosen because its rich variety of sounds, silent passages and abrupt changes.

The step Δ, which is different for each audio file, was decided according to SNR > 40 dB.

4.3.1 Semi-blind decoding

In Section 4.2.2 were defined two decoding strategies: *global noise variance* and *particular noise variance*. In this Section we show results of them both using $\hat{\mathbf{X}}$ to compute the noise variance.

Let us assume that $\mathbf{Y}$ is the watermarked audio after compression. $\hat{\mathbf{x}}$ and $\mathbf{y}$ are frequency coefficients taken from 25-28 of $\hat{\mathbf{X}}$ and $\mathbf{Y}$ respectively, and they are properly ordered according to time.

For *global noise variance* method, the statistics of channel noise are computed with:

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}\big((\hat{x}_i - y_i) - (\overline{\hat{\mathbf{x}} - \mathbf{y}})\big)^2. \tag{4.2}$$

This unique σ^2 is used to compute the **llr** of Section 4.2.2.

An alternative way, *particular noise variance* method, to compute the noise variance is

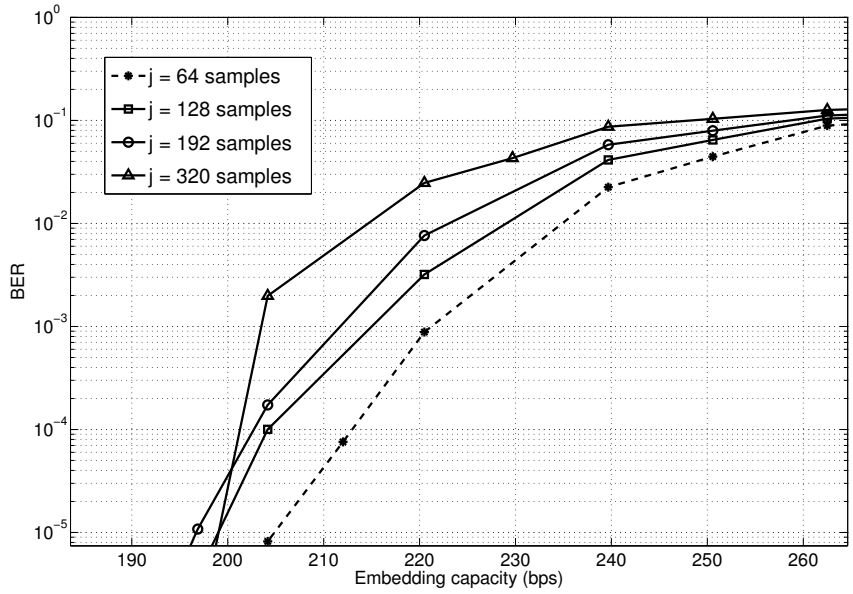


Fig. 4.5: *Particular noise variance* scheme using different number of samples j to compute the noise variance ς_p^2.

dividing the frequency coefficients $\hat{\mathbf{x}}$ and $\mathbf{y}$ in blocks $\breve{\mathbf{x}}_p$, $\breve{\mathbf{y}}_p$ and computing independent statistics of the channel noise:

$$\varsigma_p^2 = \frac{1}{j} \sum_{i=jp-j+1}^{jp} \left((\hat{x}_i - y_i) - (\overline{\breve{\mathbf{x}}_p - \breve{\mathbf{y}}_p}) \right)^2, \tag{4.3}$$

where j is the number of samples for each block $\breve{\mathbf{x}}_p$, $\breve{\mathbf{y}}_p$ and ς_p^2 is its noise variance. $\breve{\mathbf{x}}_p$ and $\breve{\mathbf{y}}_p$ are vectors with elements from $[jp - j + 1, jp]$, i.e. $\breve{\mathbf{x}}_p = \hat{x}_{jp-j+1}, \ldots, \hat{x}_{jp}$.

The next step is to decide how many samples j are needed to estimate a reliable ς_p^2. Our proposal divides the audio file in blocks $\hat{T}_h$ of 512 samples. From each time-domain block $\hat{T}_h$, 64 frequency coefficients are obtained considering that only elements from 25-28 are taken. Therefore, the number of elements j is preferable to be a multiple of 64 because in that way the system keeps relation with the division on time domain.

Fig. 4.5 is a simulation for *particular noise variance* method using different number of samples j to compute the noise variance after compression with MP3. The best performance is achieved when ς_p^2 is computed using $j = 64$ frequency coefficients which is equivalent to estimate an independent noise variance for each block $\hat{T}_h$ of 512 samples. If the number of samples j is increased the performance tends to decrease. Therefore, the best way to characterize the noise produced by MP3 is to compute independent statistics of the channel noise according to the division in time domain.

The proposed methods, *global noise variance* and *particular noise variance* using semi-blind decoding are compared in Fig. 4.6, together with another two helpful simulations. In "Full-band repetition code", the watermark was embedded in all wavelet sub-bands, from 1

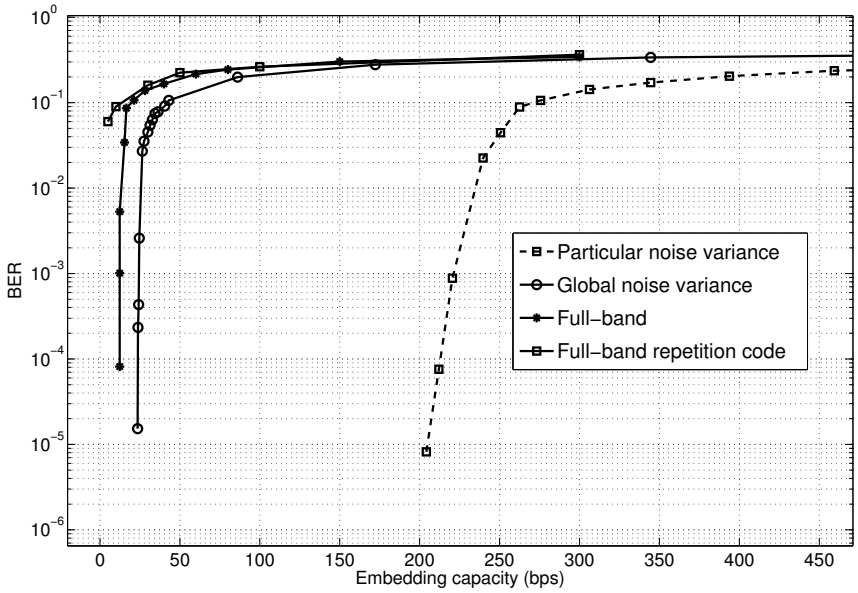


Fig. 4.6: Performance of the proposed method using semi-blind decoding.

to 32, and only repetition codes were used. The performance is the worst and we were unable to obtain lower BER.

"Full-band" uses a concatenation of outer LDPC code with inner repetition code to encode the watermark. The coded-watermark was embedded in all wavelet sub-bands. With this method low BER was achieved but the performance is still poor. The LDPC decoder uses semi-blind estimation of the noise variance.

"Global noise variance" is the method described in Section 4.2.2, the watermark is only embedded in 25-28 achieving an embedding capacity of 23.56 bps.

Finally "Particular noise variance", proposed in Section 4.2.2, is the best method. In this case, the watermark was embedded in 25-28 as well. The payload is 204.2 bps with probability of error lower than 10^{-5}. This method embeds 8.2 more bits per second than the proposal in [41].

4.3.2 Blind decoding

In the previous Section the watermarked audio $\hat{\mathbf{X}}$ is needed to estimate the noise. However blind decoding systems are desirable because audio files, especially in WAVE format, need considerable storage space. Therefore, we propose a blind algorithm using *pilot symbols* only for *particular noise variance* method which was the best method from Section 4.3.1.

In *pilot symbols* approach, a few bits are known for both encoder and decoder. Thus, the decoder is capable to estimate the noise variance with those bits.

The pilot bits are multiplexed with the encoded-watermark $\bar{\mathbf{m}}$ and embedded in the audio file. We have seen that *particular noise variance* performs better, therefore the noise variance

must be computed individually for each block in time domain.

Let us assume that T_h represents a block of 512 audio samples in time domain. After wavelet packet transformation of T_h and gathering only coefficients from 25-28, a block F_h with 64 frequency coefficients is obtained. The coded-watermark $\bar{\mathbf{m}}$ is divided in blocks W_h of $64-j$ elements. Then, j pilot bits are generated in pseudo-random way using a secret key. The block W_h is multiplexed with the j pilot bits and the result is embedded in F_h using DM. The output of this process will be the watermarked frequency coefficients which include the watermark and the pilot bits, Fig. 4.7.

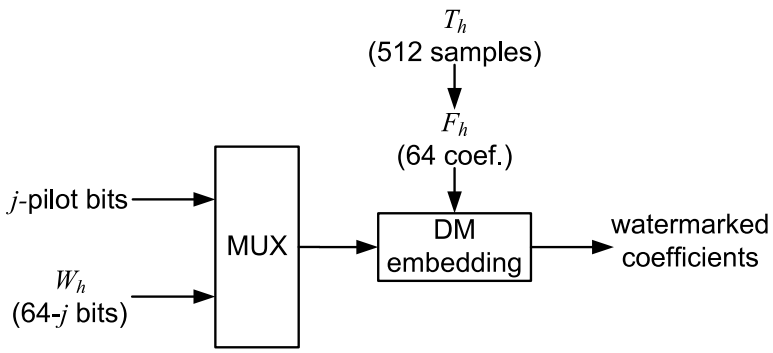


Fig. 4.7: Multiplexing pilot bits with the coded-watermark.

The decoder uses a demultiplexer to separate the noisy version of the pilot bits and the watermark. Since the decoder has perfect knowledge of the pilot bits, and estimate ς_h^2 is computed for each block $\hat{F}_h$ and the log-likelihood ratio is obtained with (4.1).

However there is a trade-off between the number of pilot bits j and the watermark payload. That is, the more pilot bits per block F_h, the better channel estimation that results in a better decoding. But, the more pilot bits the lower redundancy of the watermark and therefore weaker watermarks.

Fig. 4.8 is a comparison of blind decoding of *particular noise variance* method using a different number of pilot bits j after MP3 at 64 kbps. With a few pilot bits, $j = 5$, per block F_h the performance is the worst. Choosing $j = 20$ produces good channel estimation, however the watermark is weak and MP3 generates many errors. The best performance is achieved when $j = 10$ pilot bits per block are used to compute the noise variance ς_h^2. The embedding capacity of this proposal using blind decoding is 155 bps.

The difference between blind decoding and semi-blind decoding for *particular noise variance* method is shown in Fig. 4.9, they differ in 49.2 bps. The gap between both methods can be reduced if more advanced techniques to compute the noise variance are implemented in the blind method. Nevertheless our blind proposal achieves similar payload than, in our knowledge, the highest payload algorithm [41] from the literature for blind audio watermarking with attacks of MP3 at 64 kbps.

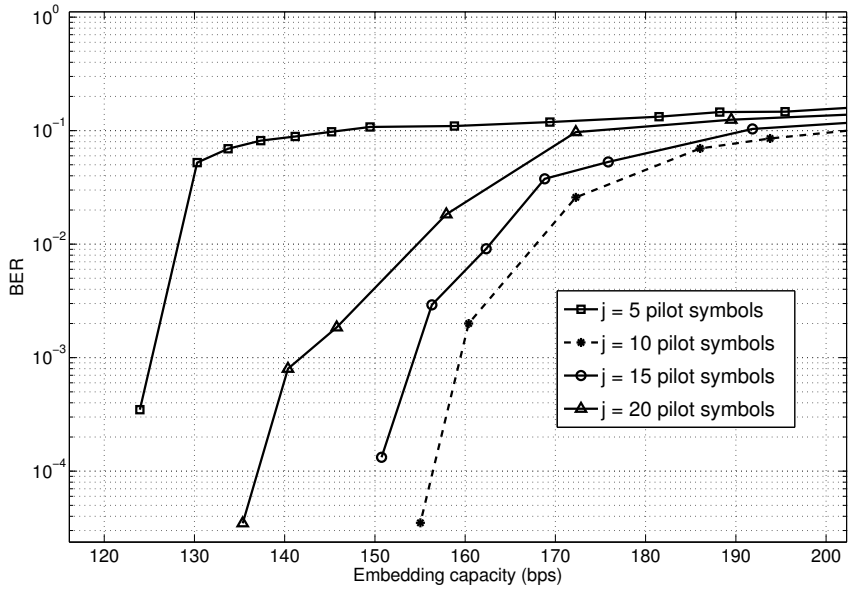


Fig. 4.8: *Particular noise variance* scheme using different number of pilot bits j per block F_h.

4.4 Increasing the embedding capacity with erasures.

Error correcting codes are capable to correct twice erasures than errors assuming that the non-erased bits are correct. An erasure means that no information about a certain bit is provided. For example, erasure of the bit $\bar{m}_i$ can be represented with $r_i = 0$.

From Section 4.2.2 and also from Fig. 4.4, we have seen that certain parts of the audio files are susceptible to produce more errors after the attack. If there is a correlation related with the errors, we can define erasures in regions with high density of noise and avoid many errors. Table 4.1 shows the correlation between the amount of noise in sub-bands 25-28 and some characteristics of the audio file. The attacks applied to the audio files were MP3 compression at 64 kbps and low-pass filtering with a cutoff of 6 kHz. High correlation is obtained between the amount of noise and the average energy of sub-bands 25-28. Based on this correlation, erasures can be defined in places with high average energy of sub-bands 25-28.

The encoding method for this proposal is the same method already explained in this Chapter. At the decoder, $\mathbf{r}$ is computed as in Section 4.2.2. For each block $\hat{T}_h$, the average energy e_h of its wavelet sub-bands 25-28 is computed. If r_i comes from a block where $e_h > t$ then $r_i = 0$ is defined as erasure, where t is a threshold. De-interleaving is applied to $\mathbf{r}$ and the rest of the decoding process is exactly as *particular noise variance*.

Using erasures, the embedded capacity can be increased by more than 15 bps in comparison with normal decoding for MP3 compression, Fig. 4.10. The final embedding capacity achieved is 229.7 bps, yet with MP3 at 64 kbps.

Since MP3 cuts the high frequencies of the audio file, we also tested the algorithm against low-pass filtering with a cutoff frequency of $f_c = 6$ kHz. The result, shown in Fig. 4.11,

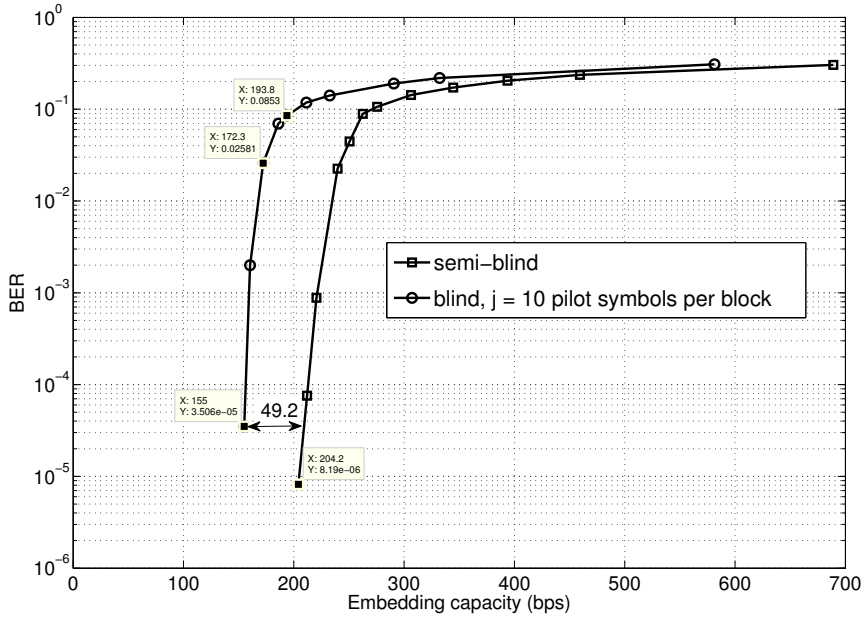


Fig. 4.9: *Particular noise variance* scheme: Performance of blind decoding versus semi-blind decoding.

confirms the good performance of our algorithm using erasures at the decoder. The benefit is of 5 bps compared with normal decoding. The embedding capacity achieved was of 115 bps with a BER lower than 10^{-5}.

Due to huge dynamic range of audio signals, we have not found a constant threshold t for audio files, even for the same audio file the best threshold is different for different watermark rates. For example, in Fig. 4.12 the best thresholds for the audio file: *A change of seasons* are shown. So far, it has been noticed that the best threshold can be found in at most 10 iterations using a brute force search in a range $t \in [.001, .05]$.

Even there is a strong noise-correlation, higher than .8, we have noticed that noisy blocks $\hat{F}_h$ only have a number of errors slightly higher than half of the total bits embedded. The reason of this behavior is that DM only does a decoding mistake when $\text{mod}\,(|\hat{x}_i - y_i|, \Delta/2) > (\Delta/4)$ and it is not strictly related with the noise strength. Therefore, when the whole noisy block $\hat{F}_h$ is defined as erasures, we are discarding erroneous information that represents slightly more than half of the embedded bits in that block, but we are also rejected a considerable amount of correct information.

Better schemes could be developed if there is a correlation between each wavelet coefficient itself and the errors. In that case, there would not be need to define the whole block as erasures, only the most likely erroneous coefficients will be defined as erasures instead.

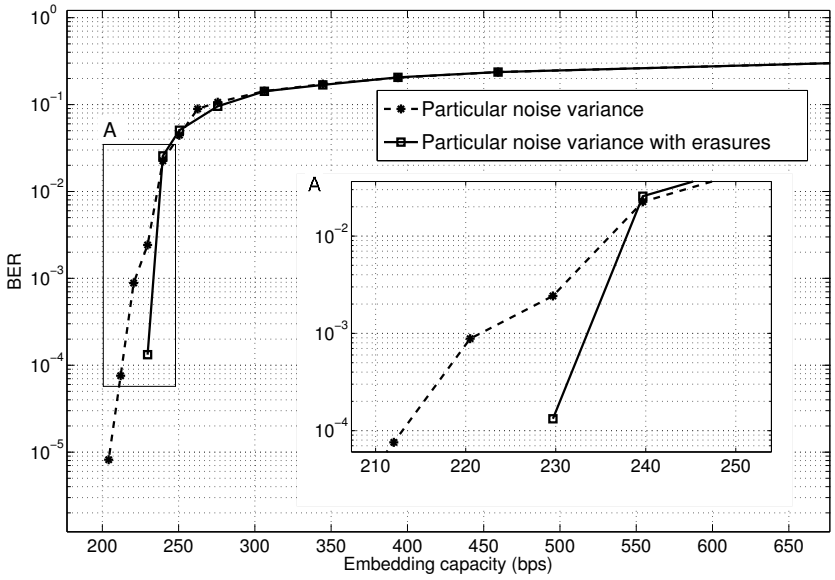


Fig. 4.10: Outperform of semi-blind *particular noise variance* scheme using erasures at decoder for MP3 compression.

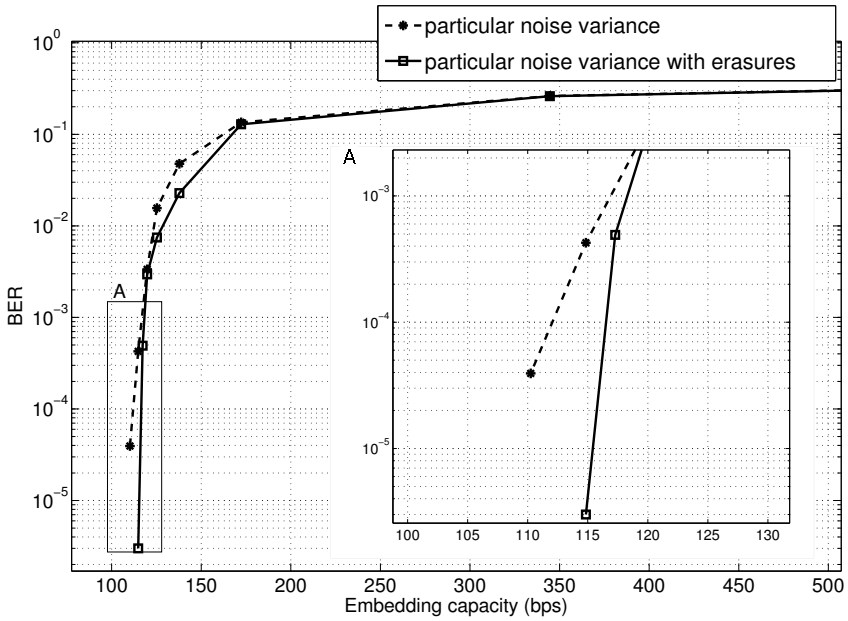


Fig. 4.11: Outperform of semi-blind *particular noise variance* scheme using erasures at decoder for low-pass filtering.

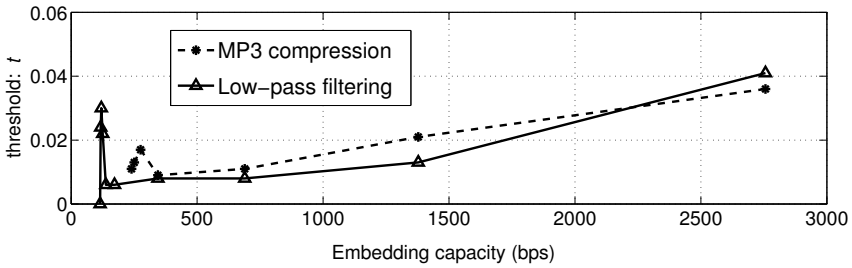


Fig. 4.12: Best threshold t for different embedding rates.

Table 4.1: Correlations related with the amount of noise in wavelet sub-bands 25-28.

Amount of noise from sub-bands 25-28 and	**Value of correlation**	
	MP3 64 kbps	Low-pass 6 kHz
Variance (time)	0.193	0.170
Block energy (time)	0.193	0.171
Block av. energy (time)	0.190	0.170
Frequency (time)	0.651	0.658
Av. energy of sub-band 1-4	0.134	0.130
Av. energy of sub-band 5-8	0.572	0.568
Av. energy of sub-band 9-12	0.678	0.679
Av. energy of sub-band 13-16	0.715	0.710
Av. energy of sub-band 17-20	0.567	0.564
Av. energy of sub-band 21-24	0.609	0.690
Av. energy of sub-band 25-28	**0.819**	**0.830**
Av. energy of sub-band 29-32	0.782	0.779

4.5 Comparison with related algorithms

MP3 is a challenging attack for audio watermarking, and this fact is reflected in Table 4.2 where 71% percent of the reported algorithms have a payload lower than 100 bps. In some cases, e.g. [60] and [67], the authors claim that their algorithms are robust to compression but they did not report the BER.

Algorithms that overcome the barrier of 100 bps are reported in [64], [40], [68] and [41]. However the achieved payload, by [64] and [40], is not tolerant to compression of 64 kbps. Wu *et ál.* [68] proposed an interesting algorithm that achieves 172 bps for compression at 64 kbps with BER = 0.043. Our basic algorithm using pilot symbols achieves the same payload, 172.3 bps with a lower BER = 0.025, see Fig. 4.9.

The best algorithm, in our knowledge, reported in the literature is the scheme proposed in [41] by Bhat *et ál.* This algorithm has a payload of 196 bps and it is resistant to MP3 at 64 kbps although its BER = .01 is still high, because 1 of every 100 embedded bits is erroneous. Our basic algorithm using pilot symbol has similar performance, achieving 193.8 bps with BER = .08. Using semi-blind decoding our algorithm embeds 8.2 bps more than Bhat's algorithm with an even much lower BER = 8.19×10^{-6}, that is, around 8 erroneous bits of every million of embedded bits, refers to Fig. 4.9.

Moreover our best result, Fig. 4.10, has a benefit of 33.7 bps over Bhat's algorithm. In summary, our algorithms obtain a payload of 155 bps for the basic algorithm with pilot symbols, in Section 4.3.2; 204.2 bps for the semi-blind algorithm of Section 4.3.1, and 229.7 bps for the algorithm with erasures at decoder, described in Section 4.4.

Since the aim of this paper is not the channel estimation and for the sake of simplicity, we introduced only a basic blind decoding which uses pilot symbols. However, our semi-

Table 4.2: BER and payload comparison of related algorithms against MP3 compression.

Algorithm		**MP3 Quality [kbps]**	**BER**	**Payload [bps]**
Cvejic	[31], 2003	32	.0028	27.1
In-kwon	[58], 2003	96	.0020	10
Wang	[59], 2004	128	.0571	10.72
Wu	[68], 2005	64	.0434	172
Chang	[60], 2006	56	YES	86
Li	[61], 2006	32	.0156	4.26
Xiang	[62], 2007	128	.1500	3
Xiang	[63], 2008	64	.1750	2
Erçelebi	[64], 2008	128	.4900	170
Deshpande	[40], 2008	96	.0025	220
Fan	[65], 2009	48	.0347	86
Wang	[66], 2009	64	.0100	Not reported
Megías	[67], 2010	96	YES	30.09
Bhat	[41], 2010	64	.0100	196
Ours		64	1.323×10^{-4}	229.7

blind decoding algorithm could be easily converted to blind using a good channel estimation technique, because $\hat{\mathbf{X}}$ is only required to compute the noise variance.

4.6 Audio quality tests

Quality of the watermarked signal is important because a signal with bad quality loses its commercial value. This aspect becomes even more important in audio files because the human auditory system is more sensible to perceive changes than other senses, e.g. the sight.

All the simulations presented in this Section produced watermarked audio with an average SNR = 44.1 dB and variance of .02. However, SNR is not the most suitable metric for measuring audible distortion. Therefore, the quality of the watermarked audio is also measured with two different tests: the *ITU-R BS.1116* [72] and the *ITU-R BS 1387.1* [73] standards. The former is a subjective evaluation of small impairments of high-quality perceptual audio codecs. The latter is an objective measurement better known as *perceptual evaluation of audio quality* (PEAQ).

The subjective test, *ITU-R BS.1116*, was applied to 23 persons with three different audio files: classic, pop and rock music. The grading is defined with 5.0 to refer that the watermark is *imperceptible* from the audio file, 4.0 for *perceptible, but not annoying*, 3.0 for *slightly annoying*, 2.0 for *annoying* and 1.0 for *very annoying*. The results are shown in Table 4.3, in all of them the quality of the watermarked audio is higher than 4.0.

The last quality test was performed with PEAQ standard. This test is objective but has similar grading that the previous. A grading of 0.0 means that the watermark is *imperceptible*,

Table 4.3: Subjective evaluation of audio quality based on *ITU-R BS.1116 standard*.

Audio	**Score**
Classic: *Egmont Op. 84*	4.14
Pop: *Billie Jean* by M. Jackson	4.78
Rock: *A change of seasons* by Dream Theater	4.28

-1.0 for *perceptible, but not annoying*, -2.0 for *slightly annoying*, -3.0 for *annoying* and -4.0 for *very annoying*. The results, reported in Table 4.4, show that the watermark is almost imperceptible for *rock* and *pop* music. We have noticed that in quiet segments of the music the watermark is apparently perceptible because there is not a strong masking sound. This is reflected in the grading for *classic* music which contains many quiet segments that enables the perception of the watermark, however this perception is thin and not annoying as reported in the subjective test in Table 4.3.

Table 4.4: Objective evaluation of audio quality based on *ITU-R BS 1387.1 standard*, PEAQ.

Audio	**Score**
Classic: *Egmont Op. 84*	-1.3035
Pop: *Billie Jean* by M. Jackson	-0.661
Rock: *A change of seasons* by Dream Theater	-0.411

4.7 Conclusion

High payload algorithms for audio watermarking resilient to MP3 compression have been proposed. Our algorithms embed a binary coded-watermark inside the frequency of audio files using DM. The coded-watermark is generated with a concatenation of LDPC codes with repetition codes.

Effects of MP3 on wavelet coefficients were analyzed. Coefficients from sub-bands 25 to 28 were found to suffer less distortion considering that the audio is decomposed with wavelet packet until the fifth level. Sub-bands 25 to 28 belong to middle frequencies of the audio, equivalent to frequencies from the 11 kHz until 13.7 kHz.

The statistics of the channel noise were computed using independent estimations along the audio file, producing a more accurate variance of the channel noise.

Reflections from the previous paragraphs allow us to propose an algorithm called *particular noise variance* which has an embedding capacity higher than all previous known proposals resilient to MP3 at 64 kbps. This algorithm is capable to use semi-blind and blind decoding. With semi-blind decoding, the algorithm obtained a benefit of 8.2 bps over the algorithm with the highest payload reported in literature [41] for MP3 of 64 kbps and with blind decoding, our algorithm has similar performance to [41].

A strong correlation between the amount of noise and the average energy from sub-bands 25-28 was found. This correlation was used to define erasures on places which are likely to be noisy. This idea increased the embedding capacity of the *particular noise variance* method by more than 15 bps. The final achieved payload was of 229.7 bps, which overperforms the algorithm in [41] by 33.7 bps.

Finally, the distortion in the watermarked audio was evaluated. The quality of audio was measured with three methods. The first one shows SNR higher than 40 dB. The second one was a subjective evaluation with more than 20 persons. The results showed a score higher than 4 of 5 possible for the watermarked audio, where 5 represents the best audio quality. The last one is an objective measurement based on the *ITU-R BS 1387.1 standard* better known as "PEAQ" and showed that the watermark is imperceptible for rock and pop music while perceptible but not annoying for classic music.

5

Self-synchronous decoding with LDPC codes

Reliable communications through noisy channels must involve techniques to prevent errors generated in the transmissions. Block error-correcting codes are algorithms that attempt to correct those errors by dividing the information into blocks and adding redundancy to each block. However the encoder and the decoder must be in perfect synchrony, that is, they must know exactly where a certain block begins and ends. Otherwise, even with the lack of noise, the decoder will emit unintelligible results. Hence, synchronization is important when the information is transmitted as blocks of bits called codewords.

Synchronization can be lost due to many reasons. For example, clipped codewords produced by deletion of symbols in the channel, interruption of the communication between transmitter-receiver, carrier phase offset, etc.

One of the first attempts to keep the synchronization was the use of a special symbol which delimited the end and the beginning of different codewords, with the constraint that the special symbol was not a member of the information alphabet. Morse code applies this technique, and the letter space is the delimiter among codewords.

Codeword separation is implicit on comma-free codes [74] and [75]. These codes have the property that no valid codeword is a prefix of any other valid codeword. Therefore, if the received stream is read from left to right the codeword is immediately recognized because it is the only block that looks like valid codeword. However comma-free codes do not have such good error-correction capability as other codes do. In addition, if the code length is large, the decoding produces delays.

In 1965, Levenshtein proposed binary codes which aided maintaining the synchronization [76]. His aim was not codeword separation, he defined a code that is capable of recovering codewords with s or fewer deletions/insertions instead. However, depending on the code,

some restrictions must be applied. For example, in some cases the code is only capable of correcting one deletion from every three codewords.

Popular codes used to recover the synchronization are cyclically permutable (CP) codes proposed by Gilbert [77]. Generic CP codes do not have implicit synchronization capability by themselves, but the synchronization property appears if there are redundant codewords at the decoder. An example of this principle was applied by Kuribayashi [78] in watermarking to endure desynchronization attacks. Generally, CP codes are constructed by discarding codewords from a cyclic code [77] and [79]. Therefore, CP codes have a low coding rates which is its major disadvantage. There is a subset of CP codes that are also comma-free codes, however CP code does not imply comma-free. A formal definition of comma-free codes and CP codes can be found in Section 5.1.

Most of the algorithms with self-synchronous capability are tied to some specific code, e.g. CP codes, comma-free, Levenshtein's codes. However powerful error correcting codes, e.g. low-density parity-check (LDPC) codes or turbo codes, are desirable in noisy channels like the ones found in watermarking.

In [80] and [81], LDPC codes have been already used in schemes that keep the transmitter and the receiver in sync. However, the desynchronization problem is from a different nature. They both assume desynchronization because carrier phase offset which differs from the aim of our proposal. They do not consider loss of synchronization because of clipped codewords at decoder.

We propose, in this Chapter, a novel algorithm that recovers the synchronization using LDPC codes. We consider scenarios where the desynchronization is lost due to deletions in the channel that produce clipped codewords at the receiver. This problem is common in digital watermarking where attacks like clipping causes desynchronization [82], however we introduce and describe the proposal from a general point of view because the algorithm can be applied to any transmission which uses iterative decoders. The problem is described and formulated in Section 5.2 and some useful definitions are stated in Section 5.3.

In Section 5.4, we describe the self-synchronous algorithm [83] and [84] with addition of the fundamentals behind the synchronization capability, its complexity and restrictions. The algorithm assumes a received bit stream with clipped codewords. Thus, the decoder achieves synchronization with the nearest unclipped codeword. The foundations about our algorithm are based on the good *CP-characteristics* of the codes, i.e. *CP-distance* different from zero or large code length. The algorithm produces several sequences as potential codewords, then it decides the synchronized codeword based on the fact that only the synchronized codeword exists in the codebook. Theoretical results about these foundations are presented in Section 5.4.1.

The algorithm can recover the synchronization with any code with good CP-characteristics

including large and small codes. Our scheme loses around 3 dB, in comparison with synchronized transmission, for a very short LDPC code but the algorithm recovers the synchronization without pilot symbols. We show that the difference in performance between synchronized transmission and our algorithm is reduced when the code length is increased. For example, with a half-rate LDPC code with length 504 our algorithm performs equal than synchronized transmission at BER of 10^{-6}. Moreover, the algorithm is also capable to recover synchronization in scenarios without redundant codewords. All these simulations are shown in Section 5.5.

We introduce an idea about the performance bound of the proposed algorithm, in Section 5.6, with the aid of maximum likelihood decoding. An interesting point about the error bound analysis is the generation of a code which is non-linear but with cyclic-like characteristics.

In Section 5.7, we engage in some discussion related with the CP-distance and its repercussion on the performance.

Finally, our proposal is not restricted to a specific code, e.g. CP codes. Indeed, a wide variety of codes can be used including non CP codes. Section 5.8 concludes the Chapter.

5.1 Codes for synchronization

In binary transmission, a synchronization code is sequence of bits used to delimit frames of information. Since the information is also represented with bits, the synchronization code is a unique sequence which has very low probability to appear inside the information bits. This technique is widely used because its simplicity, however it is not efficient because the synchronization code does not contain any information from the source and therefore the payload is reduced.

In this Section two fundamental codes that have implicit synchronization capabilities, that is, they do not need any extra-bits to recover the synchronization but the codeword itself, are described.

5.1.1 Comma-free codes

Comma-free codes are self-separable codes that do not need any header or prefix to distinguish between two different codewords [74]. A comma-free code includes only codewords that are not prefix of any other valid codeword. Thus, the transmitted codewords are the only blocks which look like a codeword and therefore can be decoded without compare them with any other codeword.

Definition 5.1 (Comma-free code)**.** For any two different codewords $C^i = (c_1^i, c_2^i, \ldots, c_m^i)$

and $C^j = (c_1^j, c_2^j, \ldots, c_n^j)$ of length m and n, respectively, in a codebook $\mathscr{C}'$,

$$c_t^i, \ldots, c_m^i c_1^j, \ldots, c_{t-1}^j \notin \mathscr{C}', \quad t \in [2, n];$$

then, $\mathscr{C}'$ is referred as a comma-free code.

5.1.2 Cyclically permutable codes

Let us define $C = (c_1, c_2, \ldots, c_n)$ as a codeword of length n. Thus, one-bit cyclic-shift named "cyclic equivalent permutation" RC of the codeword C is:

$$RC = (c_2, c_3, \ldots, c_n, c_1),$$

in the same way we find that C contains n cyclic equivalent permutations, $RC, R^2C, \ldots, R^nC$, where $C = R^nC$.

In 1963, Gilbert [77] defined an error correcting code called cyclically permutable (CP) code. The main characteristic of this code is that any valid codeword cannot be obtained by cyclically permuting another valid codeword.

Definition 5.2 (Cyclically permutable (CP) code)**.** CP code is a set of codewords of length n such that for any two valid codewords $C^i = c_1^i, c_2^i, \ldots, c_n^i$, $C^j = c_1^j, c_2^j, \ldots, c_n^j$ and considering $t \in [1, n-1]$:

$$C^i \neq R^t C^i,$$

and for any j, where $i \neq j$,

$$R^t C^i \neq C^j.$$

Gilbert [77] and Kuribayashi [79], individually, proposed constructions of CP codes based on cyclic codes. However these codes have a disadvantage from the point of view of coding rate because CP codes are constructed by discarding cyclic equivalent codewords from the cyclic codebook.

The synchronization characteristics of CP codes arise when the same codeword is iteratively transmitted. Thus, the receiver does not need to recover synchronization, the message can be decoded from any successive n symbols instead, where n is the code length. Kuribayashi *et ál.* applied the previous characteristic to synchronization in digital video watermarking [78].

5.2 Problem definition and statements

Consider transmissions where codewords C^i from a codebook $\mathscr{C}$ are sequentially and iteratively transmitted $p > 1$ times each of them, i.e. the same codeword C^i is continuously

transmitted p times before a different codeword C^j could be transmitted:

$$\ldots, C_1^i, C_2^i, \ldots, C_p^i C_1^j, C_2^j, \ldots, C_p^j, \ldots,$$

and every codeword $C = (c_1, c_2, \ldots, c_n)$ contains n bits.

Assume loss of synchronization due to any consecutive deletion of α bits. Therefore, the decoder obtains a stream that contains a clipped codeword followed of an unclipped codeword,

$$\ldots, c_{\alpha+1}^i, c_{\alpha+2}^i, \ldots, c_n^i, c_1^i, c_2^i, \ldots, c_n^i, \ldots,$$

thus the decoder does not know where the unclipped codeword begins. The problem is to recover the synchronization with the nearest, unclipped codeword.

Deletions could occur among same codewords, $C_1^i, C_2^i, \ldots, C_p^i$, or different ones $C^i, C^j, \ldots$. However due to the sake of simplicity, we will focus firstly in the former case and then we will generalized the algorithm to include the latter case. Any other deletion pattern, e.g. a bit deletion per each codeword, is out of the scope of this algorithm.

This problem can be found in watermarking, especially for audio files. A coded binary stream, called watermark, is embedded sequentially in the audio file. However the length of the audio file might be changed due to compression or clipping attack. For example, MP3 encoders add samples at the beginning of the audio file. On the other hand, clipping attacks delete audio samples. The changes of length in the audio file produce desynchronization because misalignments of codewords or clipped codewords when the watermark is retrieved.

5.3 Preliminaries

Consider two codewords, C^i and C^j of length n them both. *Hamming distance* $d_H(C^i, C^j)$ is defined as the number of bits in which they differ. If the cyclic permutations of the codewords are considered, define *cyclic Hamming distance* as:

$$D_H(C^i, C^j) = \min_x d_H(C^i, R^x C^j),$$

where $R^x C^j$ means that the codeword C^j has been cyclic permuted x bits

Definition 5.3 (CP-distance)**.** In a code (n, k) with 2^k codewords and code length n the *CP-distance* S is defined:

$$S = \min_{i \neq j} D_H(C^i, C^j). \tag{5.1}$$

Consider any codeword $C^i = (c_1^i, c_2^i, \ldots, c_n^i) \in \mathscr{C}^c$. If all its cyclic permutations $R^t C^i$ are also valid codewords, then $\mathscr{C}^c$ is named *cyclic code*.

Definition 5.4 (Cyclic Code)**.** The linear code $\mathscr{C}^c$ of length n is a cyclic code if it is invariant under a cyclic permutation:

$$C^i \in \mathscr{C}^c \Leftrightarrow R^t C^i = C^j \in \mathscr{C}^c, \text{ for } t \in [1, n).$$

Then, a cyclic code contains all n cyclic permutation of any codeword.

Proposition 1 (CP-distance of cyclic codes)**.** The CP-distance of a cyclic code $\mathscr{C}^c$ is zero.

Proof. By Definition 5.4, every codeword C^i in a cyclic code can be generated by the cyclic permutation of another codeword $R^a C^j$, that is $C^i = R^a C^j$ and therefore $D_H(C^i, C^j) = 0$. Subsequently:

$$S = \min_{i \neq j} D_H(C^i, C^j) = 0.$$

□

Proposition 2 (CP-distance of CP codes)**.** The CP-distance of a CP code $\mathscr{C}$ is different from zero.

Proof. All codewords $C^i \in \mathscr{C}$, where $\mathscr{C}$ is CP, are cyclically nonequivalent by Definition 5.2. If any two codewords $\{C^i, C^j\} \in \mathscr{C}$ are non-cyclic equivalent then $C^i \neq R^t C^j$ for $1 \leq t < n$, where n is the code length. Therefore $D_H(C^i, C^j) > 0$ and subsequently $S = \min_{i \neq j} D_H(C^i, C^j) > 0$. □

Finally, we define the minimal distance of a code d_{min} as the minimal Hamming distance between any two valid codewords $\{C^i, C^j\} \in \mathscr{C}$.

5.4 Self-synchronous decoding algorithm

The synchronization capability of the proposed algorithm is based mainly on the decoding part. The encoder is a traditional LDPC code which takes blocks of k information bits and produces codewords of n bits.

The decoder receives a desynchronized stream, $\ldots, \tilde{c}_1, \tilde{c}_2, \ldots, \tilde{c}_{2n-1}, \ldots, \tilde{c}_{3n}, \ldots$, of noisy bits. The index of $\tilde{c}_i$ could or could not match with the index of a certain codeword, i.e. $\tilde{c}_1$ does not necessarily mean the beginning of a codeword. The decoding algorithm aims to synchronize with the nearest unclipped codeword. Fig. 5.1 depicts this idea.

The decoder assumes desynchronization and takes the first $2n-1$ symbols from the noisy received stream. Those symbols are divided into n sequences $\tilde{C}^i$ of length n, each sequence differs from the previous one in a shifted bit to the right.

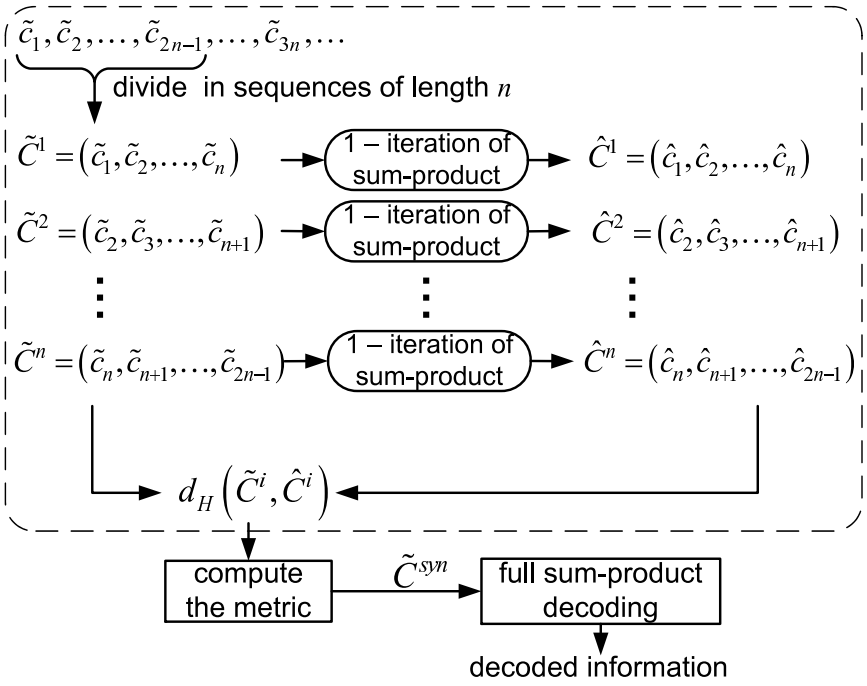


Fig. 5.1: Decoding with self-synchronization capability.

Sum-product algorithm (SPA) with only one iteration is applied to each sequence $\tilde{C}^i$ producing updated sequences $\hat{C}^i$. Hamming distance $d_H(\tilde{C}^i, \hat{C}^i)$ between the updated sequences $\hat{C}^i$ and its corresponding $\tilde{C}^i$ is measured.

With long codes, the Hamming distance $d_H(\tilde{C}^i, \hat{C}^i)$ provides reliable information about which sequence is the synchronized codeword. Fig. 5.2 shows the Hamming distances $d_H(\tilde{C}^i, \hat{C}^i)$ between potential codewords and its counterpart after one iteration of SPA decoding for a half-rate LDPC code with code length of 816. In the horizontal axis, negative and positive numbers represent desynchronized codewords shifted to the left and to the right, respectively; and zero represents the synchronized codeword. The simulation was done with different intensities of Gaussian noise SNR= $\{0, 1.5, 3.0, 5.0\}$ dB over BPSK channel. In most of them, the synchronized codeword obtained the minimum Hamming distance. When SNR= 0 dB, the noise is very high and the decoder cannot decode the correct codeword. SNR is defined as the ratio between noise power and carrier power.

However for codes with poor CP-characteristics or short codes, the Hamming distance is not enough because sometimes the minimum Hamming distance belongs to an invalid codeword. Hence, how reliable is the Hamming distance $d_H(\tilde{C}^i, \hat{C}^i)$ can be associated with the number of parity checks violated by $\hat{C}^i$. Thus, the number of invalid parity checks $\mathcal{B}^i$ from each sequence $\hat{C}^i$ is computed as well.

A simple way to combine the Hamming distance and the number of violated parity checks is to add them. Then the metric $\mathcal{M}_i$, which decides whether a sequence $\tilde{C}^i$ is synchronized,

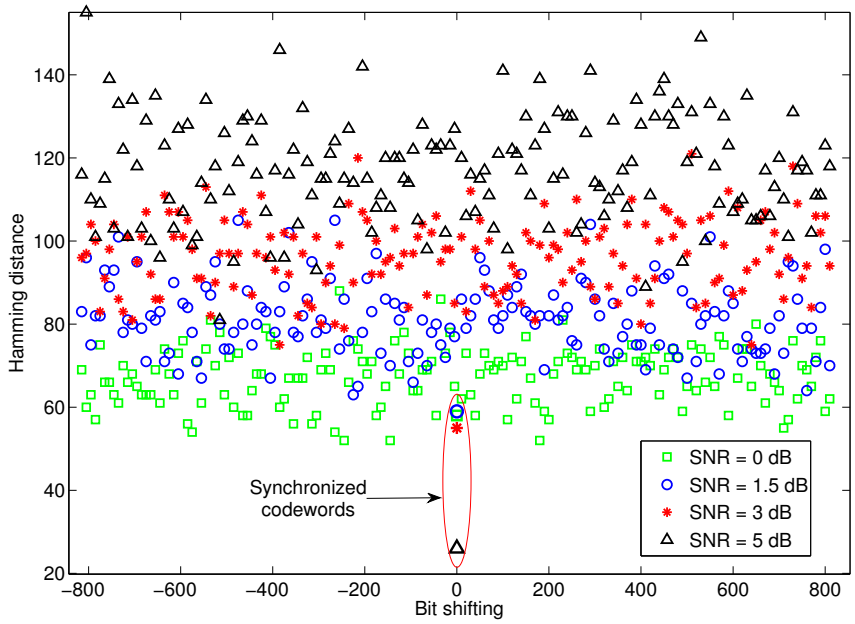


Fig. 5.2: Hamming distance $d_H(\tilde{C}, \hat{C})$ of potential codewords and its counterpart after one iteration of sum-product decoding.

is computed with:

$$\mathcal{M}_i = d_H(\tilde{C}^i, \hat{C}^i) + \mathcal{B}^i. \tag{5.2}$$

Finally, the synchronized codeword is assumed to be the sequence which minimizes,

$$\tilde{C}^{syn} = \min_i \mathcal{M}_i,$$

the metric. A basic example that illustrates the decoding algorithm can be found in Example 5.1. We refer to this algorithm as *low complexity* and it is our main proposal of this Chapter. We will propose a slightly modification to *low complexity* which produces an improvement of performance but with a rise of complexity.

Once the main algorithm was proposed, we introduce a modification in the decoding procedure to increment the performance, this second algorithm is referred as *tree* because the potential codewords are discarded in an inverted tree fashion. Each of the algorithms cover a different constraint, in *low complexity* the most important is the computational cost and in *tree* the performance is what matters. However we consider *low complexity* the main algorithm because the performance-complexity benefit is better.

The approach *tree* is shown in Fig. 5.3. The algorithm starts from the top, each node represents a potential codeword $\tilde{C}^i$ and in the top layer there are n nodes, where n is the length of the code. Every up-down transition represents an iteration of sum-product. The nodes which go to next lower layer are chosen according to the minimum value of the metric, $\min_i \mathcal{M}_i$. If there is more than one minimum for the metric $\mathcal{M}_i$, e.g. $\mathcal{M}_i = 0$, then all

Example 5.1 Recovering synchronization: noise free

Let us consider the next CP codebook $\mathscr{C}$:

$$0\,0\,0$$
$$1\,1\,0.$$

Assume that the next stream was transmitted:

$$1\,1\,0 \quad 1\,1\,0 \quad 1\,1\,0,$$

then desynchronization occurs because deletion of the first two bits producing

$$0 \quad 1\,1\,0 \quad 1\,1\,0.$$

According to the algorithm, the first $2n-1 = 2(3)-1 = 5$ bits are taken $0\,1\,1\,0\,1$ and divided into sequences $\tilde{C}^i$ of $n=3$ continuous bits:

$$\tilde{C}^1 = 0\,1\,1$$
$$\tilde{C}^2 = 1\,1\,0$$
$$\tilde{C}^3 = 1\,0\,1.$$

Each sequence $\tilde{C}^i$ is updated with SPA producing $\hat{C}^i$.

$$\begin{array}{lll} \tilde{C}^1 = 0\,1\,1 \neq \hat{C}^1 & & d_H(\tilde{C}^1, \hat{C}^1) > 0,\ \mathcal{B}^1 \geq 0 \\ \tilde{C}^2 = 1\,1\,0 = \hat{C}^2 = 1\,1\,0 & \therefore & d_H(\tilde{C}^2, \hat{C}^2) = 0,\ \mathcal{B}^2 = 0 \\ \tilde{C}^3 = 1\,0\,1 \neq \hat{C}^3 & & d_H(\tilde{C}^3, \hat{C}^3) > 0,\ \mathcal{B}^3 \geq 0 \end{array}$$

Since $\{\tilde{C}^1, \tilde{C}^3\} \notin \mathscr{C}$, hence $\tilde{C}^1 \neq \hat{C}^1$ and $\tilde{C}^3 \neq \hat{C}^3$ because SPA attempts to correct the errors. Thus, $\tilde{C}^2$ is the only sequence with metric $\mathcal{M}_2 = d_H(\tilde{C}^2, \hat{C}^2) + \mathcal{B}^2 = 0$ equal to zero and therefore $\tilde{C}^{syn} = \min_i \mathcal{M}_i = \tilde{C}^2$, $i \in [1, 3]$, is the correct and synchronized codeword.

nodes with minimum metric go to the next layer. Otherwise, if there is only one node which minimize the metric, the second minimum of $\mathcal{M}_i$ is also included in the next layer. Every time that a new layer is defined the children nodes are compared with its parent node, if a child node has the same $\mathcal{M}_i$ value as its father, that node is discarded. The process continue until obtain only one node that represents the synchronized codeword $\tilde{C}^{syn}$.

5.4.1 Fundamentals of synchronization capability

Let us think about the input segment of $2n-1$ bits which is processed by the decoder. Assuming that desynchronization occurred between the same codewords, then the segment of $2n-1$ contains the codeword $C^i = (c_1^i, c_2^i, \ldots, c_n^i)$ concatenated with a prefix $C_{pre}^i = (c_1^i, c_2^i, \ldots, c_l^i)$ or suffix $C_{suf}^i = (c_{t+1}^i, c_{t+2}^i, \ldots, c_n^i)$ of the same codeword C^i, for $\{l, t\} \in [1, n)$. That is, the decoder could obtain one of the next concatenations: $[C_{suf}^i, C^i]$, $[C^i, C_{pre}^i]$ or

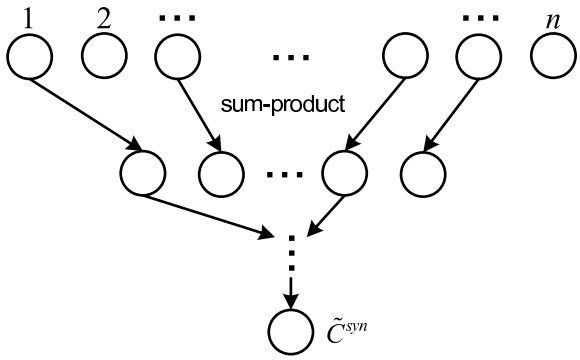


Fig. 5.3: Decoding approach following an inverted *tree* technique.

$[C^i_{suf}, C^i, C^i_{pre}]$. Without distinction about the kind of concatenation, all the n-tuples inside that segment are cyclic equivalent codewords R^tC^i of C^i. This claim can be proved with Theorem 5.1.

Theorem 5.1 (Cyclically equivalent codewords)**.** Any sequence of n consecutive bits taken from the concatenation of $[C_{suf}, C]$, $[C, C_{pre}]$ or $[C_{suf}, C, C_{pre}]$ is a cyclic equivalent codeword of $C = (c_1, c_2, \ldots, c_n)$, where C has length n and $C_{suf} = (c_{t+1}, c_{t+2}, \ldots, c_n)$, $C_{pre} = (c_1, c_2, \ldots, c_l)$ are its suffix and prefix respectively with $\{t, l\} \in [1, n)$.

Proof. A cyclic permutation R^tC of a codeword $C = (c_1, c_2, \ldots, c_n)$, where $1 \leq t < n$, is defined as:

$$R^tC = (c_{t+1}, c_{t+2}, \ldots, c_n, c_1, \ldots, c_t). \tag{5.3}$$

Let us analyze the first case when $[C_{suf}, C]$. If C has length n, thus its suffix is:

$$C_{suf} = (c_{t+1}, c_{t+2}, \ldots, c_n).$$

Then, the concatenation of $[C_{suf}, C]$ produces:

$$[C_{suf}, C] = (c_{t+1}, c_{t+2}, \ldots, c_n, c_1, c_2, \ldots, c_n).$$

Taking any sequence of n consecutive elements from $[C_{suf}, C]$, we obtain:

$$c_{t+1}, c_{t+2}, \ldots, c_n, c_1, \ldots, c_t,$$

which is exactly the definition of cyclic permutation (5.3) of the codeword C.

Second case, $[C, C_{pre}]$. Since C has length n, the first n consecutive bits are the codeword C itself. From the second n consecutive bits the problem turns to be:

$$[C_{suf}, C_{pre}] = (c_{t+1}, c_{t+2}, \ldots, c_n, c_1, c_2, \ldots, c_l),$$

and

$$C_{pre} = c_1, c_2, \ldots, c_l,$$

with $1 \leq l < n$. Considering that C_{suf} has $n - t$ elements, any consecutive string of n inside $[C_{suf}, C_{pre}]$ is defined as:

$$c_{t+1}, c_{t+2}, \ldots, c_n, c_1, \ldots, c_t,$$

which is again the definition of cyclic permutation (5.3).

In the last case when $[C_{suf}, C, C_{pre}]$, there cannot be a sequence of n consecutive bits which contains elements of both C_{suf} and C_{pre} at the same time, because there is a codeword C of length n between them. Therefore, this problem can be broken in two sub-problems: $[C_{suf}, C]$ and $[C, C_{pre}]$ which were already proved. □

If the encoder uses CP codes then no cyclic equivalent codeword $R^t C^i$ of C^i exists in the codebook. Thus, after one iteration of SPA all the updated sequences of $\tilde{C}^i$, which are $R^t C^i$, will be changed but only $\hat{C}^{syn} = \tilde{C}^{syn}$ remains the same. Therefore, $d_H(\tilde{C}^i, \hat{C}^i)$ will be zero only when $i = syn$, that is the synchronized codeword. Following the same principle for transmission over noisy channels, the codeword $\tilde{C}^i$ with $\min_i d_H(\tilde{C}^i, \hat{C}^i)$ is expected to be the most likely synchronized codeword.

The larger the CP-distance of a code, the larger the difference between $d_H(\tilde{C}^{syn}, \hat{C}^{syn})$ and $d_H(\tilde{C}^i, \hat{C}^i)$ for $i \neq syn$. Therefore the larger CP-distance, the better synchronization characteristics because $\tilde{C}^{syn}$ has lower probability to be confused with any other $\tilde{C}^i$ for $i \neq syn$.

When the code is not precisely a CP code, it must have good CP-characteristics or a long code length. Good CP-characteristics mean that the code must have the less possible number of equivalent codewords in its codebook. For short codes, this can be achieved with equivalent codes generated by elementary row operation in its generator matrix.

With long codes is difficult to say that a code is strictly CP code because the number of codewords is huge. However this characteristic itself is a great advantage in our algorithm because the probability that any desynchronized sequence belongs to the codebook is reduced when the code length is increased. For long codes is expected that our method performs very close to perfect synchronized transmission.

It has been discussed the specific case when desynchronization happens between the same codewords. However desynchronization may also occur among different codewords, producing different concatenations, $[C^i_{suf}, C^j]$, $[C^i, C^j_{pre}]$ or $[C^h_{suf}, C^i, C^j_{pre}]$ where Theorem 5.1 does not hold. However if the code is large, the probability of errors due to an invalid sequence $R^t C^j$ turns into a codeword C^i, including $j = i$, is low. We refer to this error as *mis-decoding*.

5.4.2 Complexity

According to CP-characteristics of a code, it is possible to find the synchronized codeword $\tilde{C}^{syn}$ without the aid of the metric $\mathcal{M}_i$ but using an *exhaustive search*. Assuming that there are n potential sequences $\tilde{C}^i$ and λ is the maximum number of sum-product iterations then, the *exhaustive search* has a complexity nearly to $n \cdot \lambda$ iterations because desynchronized sequences never converge. However this complexity is highly reduced in our *low complexity* algorithm because it only needs at most $n+\lambda$ iterations. Table 5.1 shows a comparison of the number of iterations needed to find the synchronized codeword with *exhaustive search* and with our proposal: *low complexity* algorithm. λ was set to be 100, however is not necessary to iterate up to the maximum if the algorithm converges earlier. Two different half-rate random LDPC codes were used, code length 2640 and 816.

Table 5.1: Number of sum-product iterations needed to decode a desynchronized codeword of length 2640 and 816 using an *exhaustive search* and our proposal: *low complexity*.

SNR	Code length 2640		Code length 816	
	Exhaustive search	Proposal	Exhaustive search	Proposal
0.0	264000	2740	81600	915
1.5	263923	2716	81519	834
3.0	263905	2644	81504	819
5.0	263903	2642	81502	817

5.4.3 Restrictions

The main restrictions of this algorithm are two. The first one is related with the kind of code. By definition 5.4, a cyclic code contains the cyclic permutation of each valid codeword which is exactly what our algorithm is trying to avoid. Therefore cyclic codes are not suitable for this proposal.

The second restriction comes from the desynchronization nature. This proposal assumes that after the continuous deletion of α bits, there is a segment of $2n-1$ bits which contains the unclipped codeword C. Any other deletion pattern, e.g. one deleted bit every codeword or similar, is out of the scope of this proposal.

Let us assume a bit stream which is a concatenation of codewords $C = (c_1, c_2, \ldots, c_n)$, then all the continuous deletion can be classified in three different patterns:

1. The first one is when the deletion occurs at the middle and leaves the beginning and the end of a codeword:

$$c_1, c_t, ?_1, \ldots, ?_\alpha, c_{t+\alpha}, \ldots, c_n, c_1, c_2, \ldots, c_n$$

2. Second one is when the last part of a codeword is missed:

$$c_1, c_2, \ldots, c_t, ?_1, \ldots, ?_\alpha, c_1, c_2, \ldots, c_n$$

3. The last is when the beginning of a codeword was deleted:

$$?_1, \ldots, ?_\alpha, c_{t+1}, \ldots, c_n, c_1, c_2, \ldots, c_n$$

Theoretical analysis was described from the point of view of case 3. However if the code has good CP-distance, errors due to cases 1 and 2 are negligible. Moreover, cases 1 and 2 become case 3 if the left part of the deletions is discarded. Theorem 5.1 can be applied when the erasure (case 3) occurs between the same codewords. Therefore, all codewords in the segment $c_{t+1}, \ldots, c_n, c_1, c_2, \ldots, c_n$ are cyclic equivalent of the synchronized codeword.

5.5 Performance over BPSK transmissions with Gaussian noise

In this Section comparison between desynchronized transmissions using the proposed algorithms and perfect synchronized transmissions are shown. The channel is a binary phase-shift keying with AWGN.

Recalling, the fundamental problem is desynchronization due to clipped codewords. Therefore, the decoder should be capable to decode the next unclipped codeword, that is, recover the synchronization. Usually in digital audio watermarking, clipped attack produces few clipped codewords, nevertheless the *extreme case* will be when from every two transmitted codewords one of them is clipped. In the simulation the synchronization is recovered for the *extreme case*, that is, a clipped codeword from every two transmitted codewords. The length of the clipped bits was chosen each time randomly.

One of the advantages of this proposal is that a wide variety of codes, e.g. short or long, can be used. Fig. 5.4 shows the comparison between the proposed algorithm and synchronized transmission for a short LDPC code with rate .25, code length 12 and CP-distance $S = 2$. The simulation was conveyed by applying desynchronization to every transmitted codeword but, in this case, the desynchronization happens only among redundant codewords. The difference of performance is about 3 dB, however the synchronization was recovered with a short code and without the aid of pilot symbols. This big gap is due to the code is very short, but is expected that the performance of both methods tend to be the same with large codes. For example in Fig. 5.5, we use a half-rate LDPC code with length of 96 and the difference between synchronized transmission and our algorithm is reduced to only 1.8 dB.

The proposed algorithm is not restricted to CP codes, however attention must be paid to the CP-characteristics of the code. That is, the code might have CP-distance $S = 0$ but the

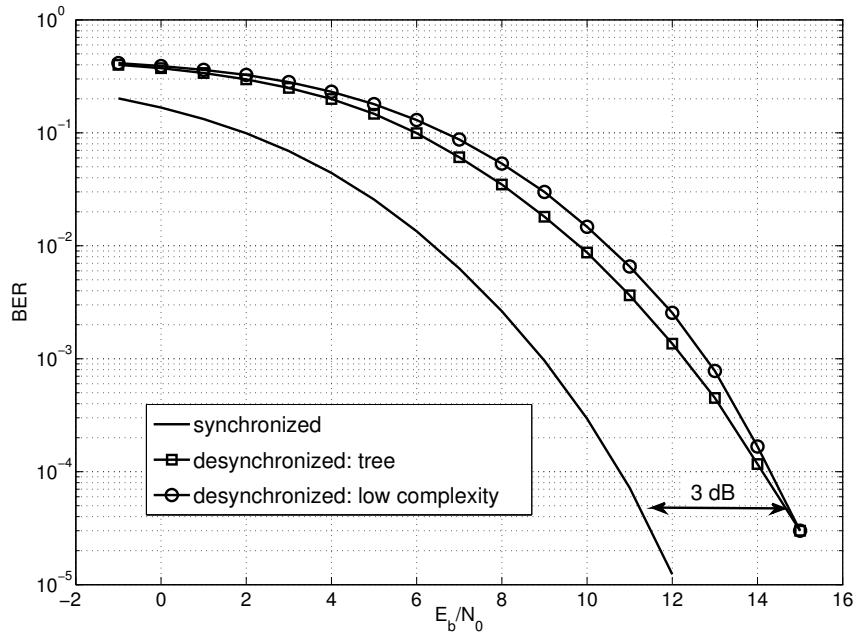


Fig. 5.4: Performance comparison between synchronized transmissions and our self-synchronizable decoding algorithm over AWGN channel using a tiny LDPC code.

code should contain a few number of equivalent cyclic codewords because those codewords produce *mis-decoding* which affects the performance of the algorithm. Fig. 5.6 shows the previous idea. In this simulation, two high rate LDPC codes were used. The first code "(a)" has length 273 and rate .69, the second one "(b)" has length 495 and rate .87. Whether these codes are precisely a CP codes or not is difficult to compute. Nevertheless, since the codes are high-rate then is highly probable that the CP-distance is zero. In this simulation, the code length is bigger than previous simulations on Figs. 5.4 and 5.5, therefore the difference between both methods has been reduced to nearly 1 dB. The deletion is still among redundant codewords. In this and further simulations only performance of *low complexity* algorithm will be shown.

In general, the proposed method works with codes of different lengths. The longer the length the better performance and less restrictions.

When the fundamentals behind this proposal were explained in Section 5.4.1, we assumed that the desynchronization occurred in places with redundant codewords. We also explained that Theorem 5.1 does not hold when desynchronization happens among different codewords. However, if the code has long code length or good CP-characteristics the probability of *mis-decoding* due to desynchronization among different codewords is low and the proposed algorithm performs as well as before.

Fig. 5.7 pictures the above situation. We used a half rate LDPC code with length 504. "synchronized" scheme is a normal transmission which was perfectly synchronized and only contaminated with AWGN. "desynchronized: A" is our proposal when the desynchronization

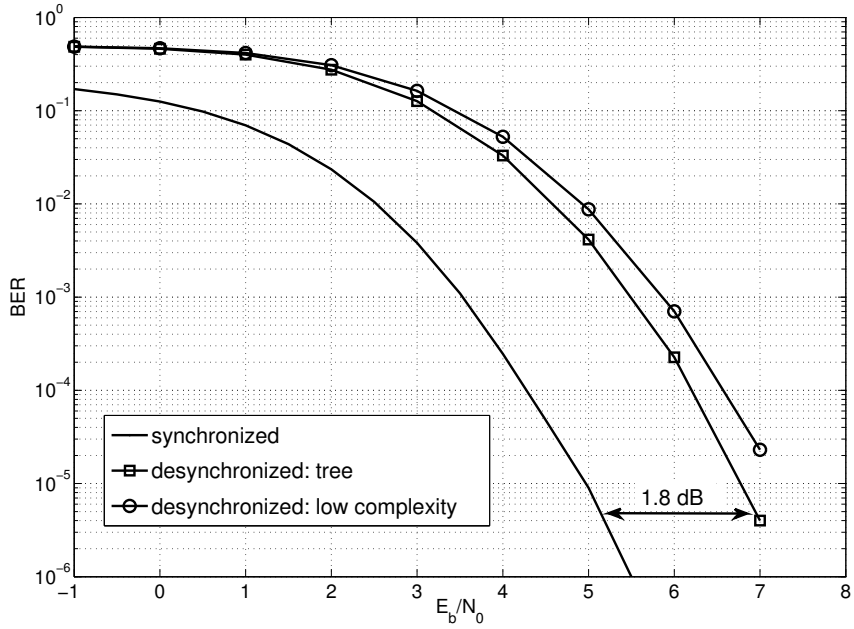


Fig. 5.5: Recovering synchronization with a small half-rate LDPC code with code length 96.

happens only among redundant codewords and "desynchronized: B" is when desynchronization occurs among different codewords. Without distinction of the desynchronization place, the algorithm performs around .75 dB less than perfect synchronized transmission for low SNR and almost the same for high SNR. Comparing the gaps between synchronized transmission and our method in Figs. 5.4, 5.5, 5.6 and 5.7, we are aware that the difference is reduced when the code length is increased.

The decoder performance relies on the CP-characteristics of the code, the larger CP-distance the better performance. Generally, large codes have good CP-characteristics, e.g. LDPC codes with codeword length around 500 or higher. LDPC cannot be considered as formal CP codes because they are constructed in random way. Nevertheless, the number cyclic equivalent codewords in its codebook is expected to be small due to the same randomness nature.

5.6 Toward the bound: Maximum likelihood decoding

The proposal's behavior has been defined and also its scopes have been discussed. However a theoretical bound that ensures the reliability of the algorithm is needed. In this Section, we introduce a simple approach based on maximum likelihood (ML) decoding which shows the bound of the error performance. Let us define $\mathscr{C}$ as the code used in our proposal. For convenience, a very small LDPC code (12,3) will be used as toy example.

Recapitulating, the proposed algorithm works under the premise that no cyclic equivalent codeword of a valid codeword C^i exists in the codebook or the number of cyclic equivalent

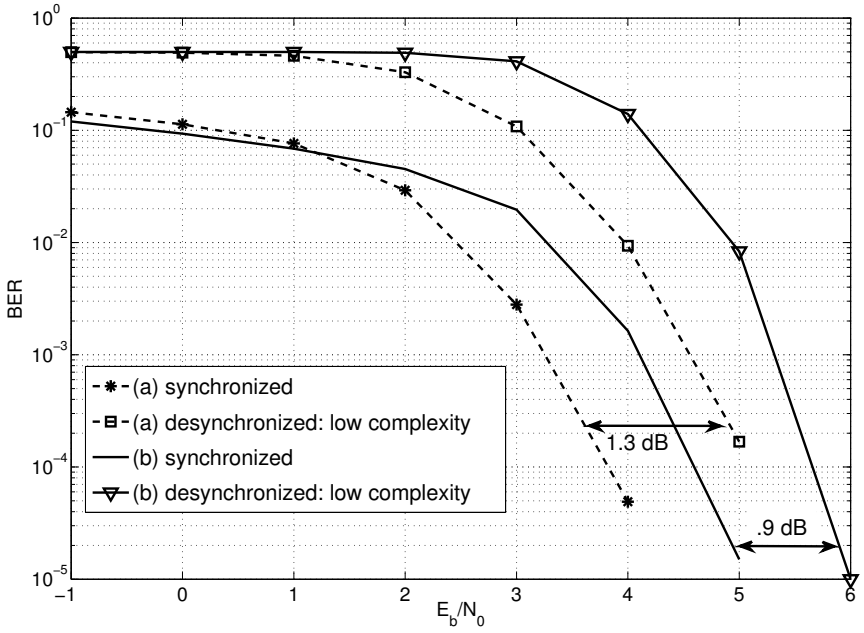


Fig. 5.6: Performance comparison between synchronized transmissions and our self-synchronizable decoding algorithm for *low complexity* over AWGN channel using a non CP codes: high-rate LDPC codes.

codewords are very few. Consider codewords $\{C^i, C^j\} \in \mathscr{C}$ including $i = j$. Thus, the algorithm produces a decoding error when either: an invalid sequence $R^t C^j$ becomes C^i or when the synchronized codeword C^i becomes a cyclic equivalent codeword $R^t C^j$ of any valid codeword in the codebook.

Now, let us think about a cyclic code $\mathscr{C}^c$. For any codeword C^i in a cyclic codebook, its cyclic equivalent codewords are also valid codewords. Then, a decoding error occurs when either any cyclic equivalent codeword $R^t C^j$ becomes C^i or vice versa, including $i = j$.

This behavior of cyclic codes under normal circumstances, i.e. synchronized transmissions, is very similar to our desynchronized model. Therefore, would be natural to think that the proposed algorithm could be bounded by a cyclic code $\mathscr{C}^c$ over a synchronized transmission, in which $\mathscr{C}$ is a subcode of $\mathscr{C}^c$.

However, it turns that the code $\mathscr{C}^c$ cannot be generated when $\mathscr{C}$ is extended, that is when all cyclic equivalent codewords of $\mathscr{C}$ are computed. A special code which is neither linear nor cyclic appears instead, we will define this code as $\mathscr{C}^*$. Therefore, well-known bounds for linear codes, e. g. *union bound*, cannot be applied.

Fig. 5.8 shows a simulation using the previous idea with a very short LDPC code, rate .25 and code length 12. "synchronized" refers to synchronized transmission using $\mathscr{C}$, "desynchronized: proposal" is the proposed algorithm with desynchronization and the codebook $\mathscr{C}$, "desynchronized: proposal ML" means the proposal with ML decoding and $\mathscr{C}$, finally "synchronized: $\mathscr{C}^*$" stands for the potential bound using codebook $\mathscr{C}^*$, synchronized, and

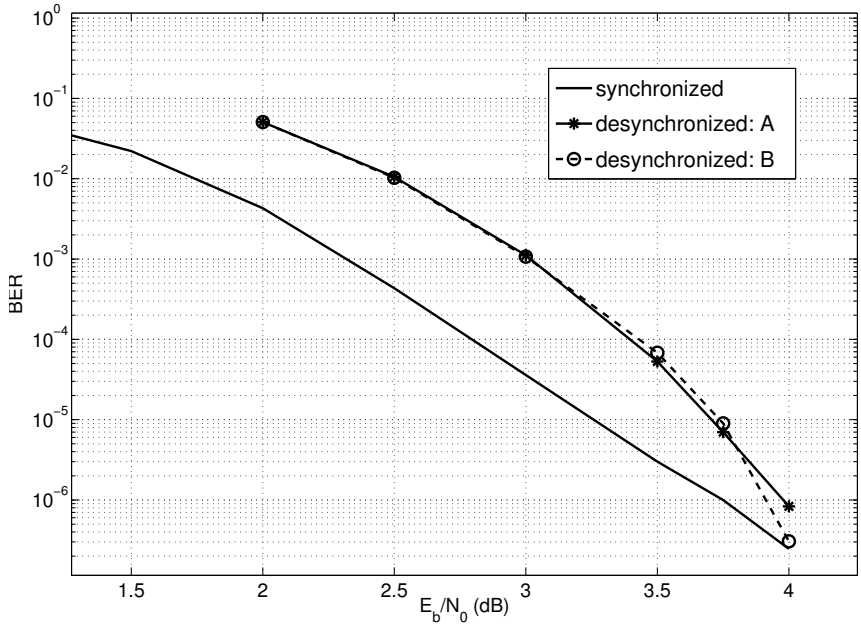


Fig. 5.7: Recovering synchronization due to, A: deletions among same codewords and B: deletions among different codewords.

with ML decoding.

This simple idea shows a good approximation about the bound of the algorithm described in this Chapter.

5.7 Codes and its CP-distance

The performance of our proposal highly relies in the CP-distance. The repercussion of different CP-distances, small and large, are analyzed in this Section.

5.7.1 Codes with large CP-distance

Codes with large CP-distance are good for synchronization. They have good performance in our scheme and its application is straightforward.

According to Theorem 5.1, all the potential codewords $\tilde{C}^i$ are cyclic equivalent but only the synchronized codeword C^{syn} exist in the codebook. The Hamming distance $d_H(\tilde{C}^i, \hat{C}^i)$ tends to be large for desynchronized codewords and small for the synchronized one. The larger is the CP-distance, the bigger is the difference between $d_H(\tilde{C}^{syn}, \hat{C}^{syn})$ and any other $d_H(\tilde{C}^i, \hat{C}^i)$ for $i \neq syn$. Therefore, synchronization with CP-codes is easily achieved.

Codes with large CP-distance belong to CP codes category. Unfortunately, according to Gilbert's [77] and Kuribayashi's [79] constructions of CP codes, these codes have a disadvantage from the point of view of coding rate. CP codes have lower coding rate than its former code because its construction is based on discarding cyclic equivalent codeword from a cyclic

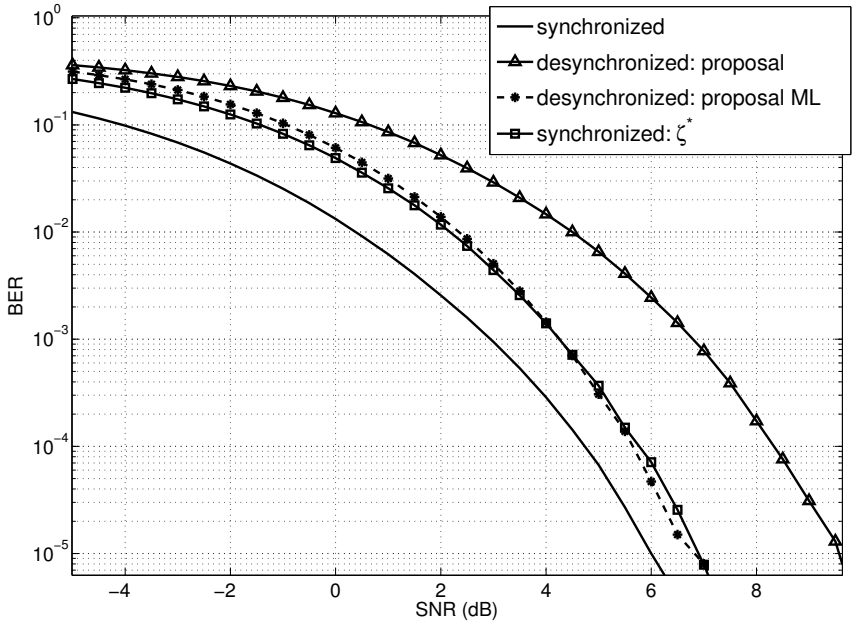


Fig. 5.8: Maximum likelihood decoding for our algorithm and its potential bounded code.

codebook. Nevertheless is always desirable to use CP codes in our algorithm if the coding rate is not a restriction.

5.7.2 Codes with small CP-distance

Small CP-distance but not zero, means that there is at least one pair of codewords in the codebook which are very similar or some of their cyclic equivalent codewords are similar. In other words, with small CP-distance is easy that a certain sequence $\tilde{C}^i$ can be turned into a valid codeword because the noise. Therefore, synchronization with codes with small CP-distance is possible but difficult.

Theoretically our algorithm cannot use codes with CP-distance zero. However further conclusions can be conveyed if the codes are analyzed separately in short and long codes. CP-distance zero means that there is at least two codewords in the codebook which are cyclic equivalent. Our algorithm decodes under the assumption that from the sequences $\tilde{C}^i$ only one exist in the codebook and all the others are cyclic equivalent. Therefore, if there are cyclic equivalent codeword in the codebook, the decoder can produce errors because cyclic equivalent codewords can be confused between each other. We already referred to this error as *mis-decoding*.

Short codes must have CP-distance different from zero to perform well with our algorithm. Short codes have a small number of codewords and therefore if the code has CP-distance zero is very likely that the decoder makes *mis-decoding* even in noise free transmission.

However, with long codes the number of valid codewords is huge. Therefore, if the number

of cyclic equivalent codewords is small, the probability of *mis-decoding* is small as well, and synchronization can be achieved even when CP-distance is zero. Fig. 5.6 is an example of the previous fact. Nevertheless the performance of the code will be affected according to the number of cyclic equivalent codewords. The algorithm can use codes with CP-distance zero as long as the code has good CP-characteristics, that is, large length and small number of cyclic equivalent codewords.

Generally small codes have also small CP-distance or even CP-distance equal to zero. However, certain small codes are desirable in specific situations due to its good error correction capability or because of constraint length in the system. Those codes must have good CP-characteristics in order to work properly with our algorithm. That is, the less number of cyclic equivalent codewords in the codebook, the better.

Small codes with bad CP-characteristics can be replaced by equivalent codes with better CP-characteristics. Equivalent codes can be generated by applying elementary row operation in the generator matrix G. For example, if we apply column permutation to a codebook with code length n, we obtain $n!$ different codebooks with different number of cyclic equivalent codewords. Therefore, is preferable to choose a code with the minimum number of equivalent codewords.

In Fig. 5.9, a simulation using two equivalent codes with the proposed algorithm is shown. First, a random LDPC code of length 20 is generated. However its performance is bad because it contains 60 cyclic equivalent codewords. Using its generator matrix, G, an equivalent code is found by elementary row operations. The equivalent code contains 0 cyclic equivalent codewords and therefore the code performs well with our self-synchronous algorithm.

5.8 Conclusion

Synchronization is a very important issue for transmission where the information is sent in blocks. In this Chapter, we described a novel self-synchronous decoding scheme for desynchronization due to continuous deletions in the channel. Besides, we extended the study by defining its scope and restrictions.

The proposal works under the assumption that the synchronized codeword has the minimum Hamming distance between the potential codewords and its updated codewords after one iteration of sum-product decoding. It was shown that the previous property holds because the CP-characteristics of the code. The longer the CP-distance of a code, the better the synchronization capability.

The results showed that the algorithm is capable to use a wide range of codes and it is not restricted to CP codes as previous proposals are. When short codes are needed, a solution was proposed by computing equivalent codes with better CP-characteristics.

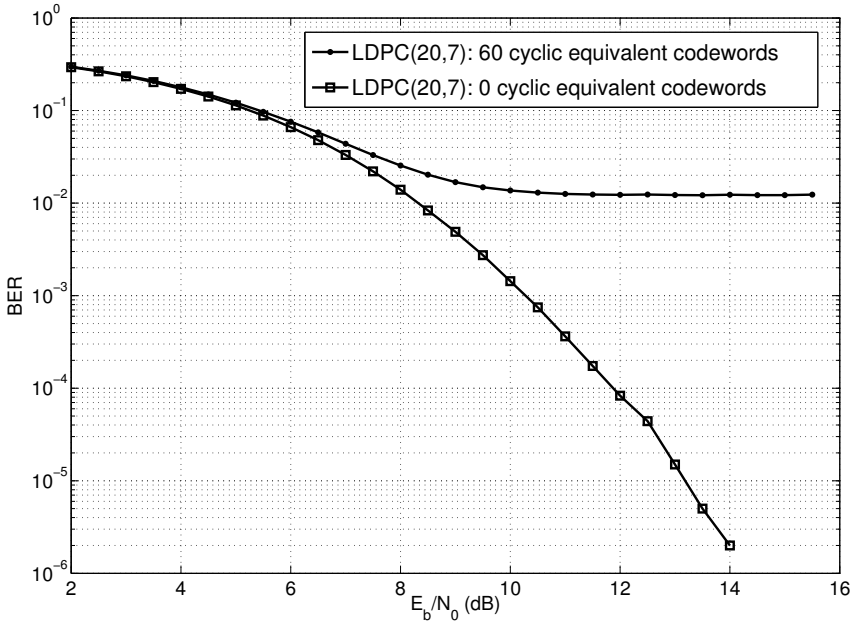


Fig. 5.9: Short codes: Performance of equivalent codes in desynchronized transmissions.

Simulation results clarify the good performance of the proposed scheme over BPSK channel with Gaussian noise. The computational complexity of the proposed algorithm is acceptable because full iterative decoding is only applied to the most likely synchronized codeword.

Two major restrictions of this algorithm are the deletion pattern and the incompatibility with cyclic codes, i.e. the algorithm cannot work with cyclic codes. The algorithm is defined only for continuous deletions, e.g. watermarking under clipping attack.

Our algorithm is capable to use CP codes and some non-CP codes. Whether CP codes are better that its counterpart non-CP is still an open question. CP codes have better performance but lower coding rate than its former non-CP code and vice versa. The previous reflection is an interesting topic to the future work.

Finally, an idea for bounding the number of errors in the algorithm was introduced. An important point is that the potential code which bounds the number of errors is neither linear nor cyclic. This idea was demonstrated with the comparison of the ML decoding of the proposed algorithm and its bounding code.

6

Self-synchronizable audio watermarking based on LDPC coding

Generally, watermarking in frequency domain produces robust watermarks [85]. Nevertheless two problems arise, desynchronization and high computational cost. Desynchronization is because the host media is clipped in time-domain and therefore is difficult to recognize the clipped part in frequency. On the other hand, even nowadays there are algorithms like the fast Fourier transform, there is not lower complexity than the native domain, e.g. time-domain.

Proposals for audio watermarking in time-domain have been described in [86] and [87]. The former is a proposal for digital instruments like synthesizers, and even the technique shows interesting results, is not suitable for a wider kind of music. The latter is a proposal from Lie *et ál.* which produces watermarks resistant to the most common audio manipulations, e.g. MPEG-1 Layer III (MP3) and low-pass filtering. Due to the algorithm was developed for time-domain, it has low computational complexity which could be good enough for practical applications working in real time. Lie's algorithm, explained in Section 6.1, is robust to clipping attack by using synchronization codes and also, the watermark was protected with convolutional coding to prevent errors due to the noise.

However, there are two disadvantages in Lie's method: the convolutional code and the synchronization code. Watermarking channels tend to obtain high BER which means than powerful coding techniques are desirable. Common techniques like BCH codes and convolutional codes practically do not improve the watermark robustness [29]. The use of synchronization codes reduces the payload because the synchronization code does not include any information about the watermark at all.

We have introduced a self-synchronous decoding algorithm using LDPC codes in Chap-

ter 5. This algorithm does not need extra-bits, e.g. synchronization codes, to keep the synchrony between the encoder and the decoder. The synchronization capability is due to the good cyclically permutable (CP) characteristics of the LDPC codes. The synchronized codeword is searched in a region which is as large as almost twice the code length. The algorithm finds the synchronized codeword after applying only one iteration of sum-product algorithm (SPA), [13], on each possible sequence.

In this Chapter, we propose an improvement to Lie's method [87] using our self-synchronous algorithm [83] and [84]. The proposed algorithm is introduced in Section 6.2. The improvements are more robust watermarks and higher embedding payload. The experimental results, in Section 6.3, show that the watermark can be correctly retrieved after combination of attacks, e.g. MP3 compression and clipping attack. Finally, conclusions are addressed in Section 6.4.

6.1 Lie's watermark algorithm

Time-domain audio watermarking can be categorized in: algorithms that modify independently each audio sample and those which modify the audio in groups of samples. The former obtains high embedding rates, however the algorithms are not robust against intentional attacks. The latter, besides producing robust watermarks, controls better the distortion in the audio file.

In this Section, the algorithm proposed by Lie *et ál.* [87] is explained. The algorithm scales the audio amplitudes in a grouping and consistent manner while embedding the watermark information. Hence, the audio envelope is almost preserved and good audio quality can be achieved. Before the embedding, the watermark is encoded with a half-rate convolutional code and, the synchrony is kept with synchronization codes which are concatenated at the beginning of each block of bits.

Definition 6.1 (GOS)**.** Group of samples (GOS) is defined as an audio segment, in time, of consecutive $\mathbf{L}$ samples. Each GOS^i is composed of three sections, sec_1^i, sec_2^i and sec_3^i, of length $\mathbf{L}^i = L_1^i + L_2^i + L_3^i$ respectively, as shown in Fig. 6.1.

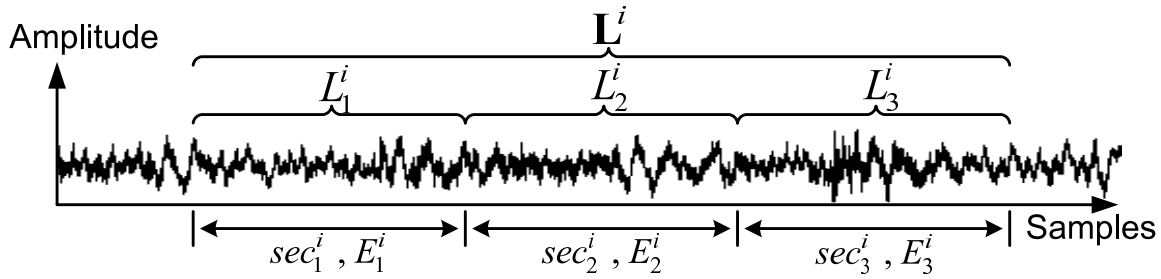


Fig. 6.1: GOS, of length $\mathbf{L}$, is composed of three different sections sec_1^i, sec_2^i and sec_3^i.

The audio file is divided into consecutive and non-overlapped GOS. Each bit will be embedded in a different GOS by modifying the average of absolute amplitudes (AOAA) of sections sec_1, sec_2 and sec_3. The AOAA E_1^i, E_2^i, E_3^i of the GOS^i are computed as:

$$E_1^i = \frac{1}{L_1^i} \sum_{j=0}^{L_1^i-1} |X_{(\mathbf{L}\cdot i+j)}|,$$

$$E_2^i = \frac{1}{L_2^i} \sum_{j=L_1^i}^{L_1^i+L_2^i-1} |X_{(\mathbf{L}\cdot i+j)}|,$$

$$E_3^i = \frac{1}{L_3^i} \sum_{j=L_1^i+L_2^i}^{\mathbf{L}^i-1} |X_{(\mathbf{L}\cdot i+j)}|,$$

where $\mathbf{X} = X_1, X_2, \ldots$ are the audio samples.

E_1, E_2, E_3 are sorted and classified in the section with the minimum, middle, and maximum energy: E_{min}, E_{mid}, E_{max} respectively. The differences, A and B, are computed with:

$$A = E_{max} - E_{mid}, \tag{6.1}$$

$$B = E_{mid} - E_{min}. \tag{6.2}$$

For a certain GOS^i, the watermark bit "1" will be represented when $A^i \geq B^i$ otherwise that GOS^i contains a "0".

The embedding is conveyed according to the next rules:

- Watermark bit "1":
  If $(A - B \geq \text{Thd})$, then no operation is performed.
  Else, increase E_{max} and decrease E_{mid} by the same amount $\rho_{(1)}$ so that the above condition is satisfied.

- Watermark bit "0":
  If $(B - A \geq \text{Thd})$, then no operation is performed.
  Else, increase E_{mid} and decrease E_{min} by the same amount $\rho_{(0)}$ so that the above condition is satisfied.

The threshold Thd is computed with:

$$\text{Thd} = (E_{max} + 2E_{mid} + E_{min}) \cdot \gamma,$$

where γ is a parameter that controls the robustness-distortion trade-off. Thus, ρ is computed

depending on the bit to be embedded:

$$\rho_{(1)} = \frac{\text{Thd} - (A - B)}{3},$$
$$\rho_{(0)} = \frac{\text{Thd} - (B - A)}{3}.$$

Finally, E_{min}, E_{mid}, E_{max} can be modified by scaling (up or down) sample amplitudes in the corresponding GOS. The watermarked audio is obtained with $\hat{\mathbf{X}} = \omega \cdot \mathbf{X}$, where $\mathbf{X} = X_1, X_2, ... \in sec_j$, $j = 1, 2$ or 3 and ω is computed with

$$\omega_{up} = 1 + (\rho/E)$$

and otherwise, to decrease we have

$$\omega_{down} = 1 - (\rho/E).$$

Initially γ is set to $\gamma = 0.05$ for each GOS^i, however not all the GOS has the same masking characteristics and therefore the watermark can be audible in some GOS. Thus, γ has to be independently tuned for each GOS^i according to the psychoacoustic model. If the distortion with the initial conditions is high, then γ is decreased by .01 and the whole embedding process is repeated until the audio quality constraint is satisfied. For more details refer to [87].

The watermark extraction is done by computing A' and B' from the watermarked audio, in the same fashion as in the embedding phase using (6.1) and (6.2). Then, if $A' \geq B'$ the bit "1" is retrieved otherwise "0" is recovered.

6.1.1 Encoding the watermark

The watermark is protected with a half-rate convolutional code (2,1,3) which takes one input bit and produces 2 output bits. The output bits depend on the last 3 input bits. At the decoder, Viterbi algorithm is used to recover the watermark.

6.1.2 Synchronization

Lie's algorithm is also robust to desynchronization produced by clipping attack. The solution is to concatenate a synchronization code at the beginning of each bit stream as shown in Fig. 6.2a. The synchronization code is a special vector of $N_1 = 20$ bits. This vector is unique and is very low likely that the synchronization code appears inside the coded watermark stream.

The decoder searches for the synchronization code in an area of $\mathbf{L} \cdot (N_1 + N_2)$ samples, where N_1 and N_2 are the numbers of bits of the synchronization code and the convolutional-

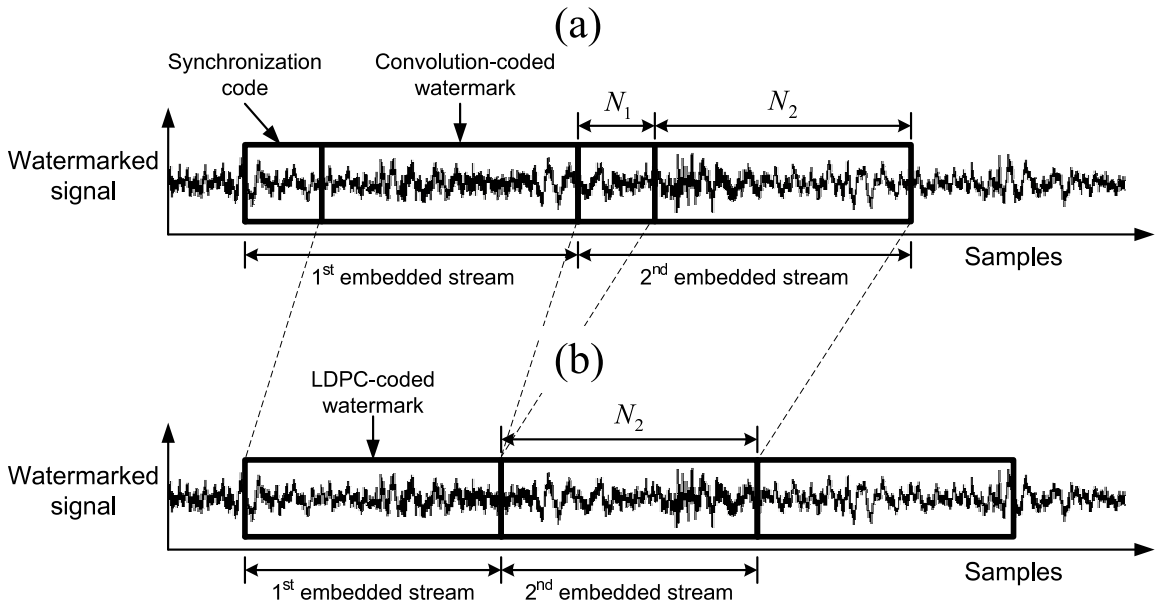


Fig. 6.2: Structure of the embedding: (a) Lie's method, concatenation of convolution-coded watermark and synchronization code. (b) Our proposal, LDPC-coded watermark with self-synchronization capabilities.

coded watermark respectively. The decoder takes a window with the first $\mathbf{L} \cdot N_1$ samples and retrieves the embedded bits, those bits are compared with the synchronization code pattern. If they do not match, the window is shifted one sample to the right and the embedded bits are retrieved again until match with the synchronization code.

6.2 Self-synchronizable audio watermarking

Lie's method, described in [87] and summarized in Section 6.1, has low-complexity and acceptable robustness e.g. against MP3 at 128 kbps. However, it has two main disadvantages: first, the synchronization code reduces the payload and second, watermarking channels tend to be very noisy and therefore convolutional codes are not the best selection.

We introduced a novel self-synchronous decoding algorithm for common transmission in Chapter 5. In this Section, we present an improvement of Lie's method by replacing the convolutional code and the synchronization code with our self-synchronous algorithm. The improved algorithm obtains higher payload than the original because it does not need synchronization codes. The robustness is also increased due to LDPC codes are more powerful codes than convolutional codes.

6.2.1 Watermark embedding

The binary watermark $\mathbf{m} = m_1, m_2, \ldots$ is encoded with an LDPC code. The coded watermark $\bar{\mathbf{m}} = \bar{m}_1, \bar{m}_2, \ldots$ is embedded in the audio file $\mathbf{X} = X_1, X_2, \ldots$ following the procedure

of Section 6.1. However, Section 6.1.1 and Section 6.1.2 are omitted; that is, the convolutional code is not used and the synchronization code is not concatenated either. Fig. 6.2b pictures this idea.

6.2.2 Watermark detection

The watermarked audio $\hat{\mathbf{X}}$ is susceptible to suffer distortions due to attacks. Therefore $\mathbf{Y} = \hat{\mathbf{X}} + \mathbf{v}$ is defined as the watermarked audio with attacks or noise $\mathbf{v}$.

When the decoder receives $\mathbf{Y}$, it does not have any knowledge about where the embedded watermark begins. Therefore, synchronization is needed. Let us assume that every coded watermark bit $\bar{m}_i$ was embedded in a GOS^i. The code length of the LDPC code is N_2. Also assume that the coded watermark was embedded continuously, that is one codeword after another. Therefore the synchronized codeword $\tilde{C}^{syn} = c_1, c_2, \ldots, c_n$ exists inside a segment $\mathbf{P}$ of $(2N_2 - 1) \cdot \mathbf{L}$ samples.

The audio segment $\mathbf{P}$ is divided into overlapped blocks $\mathbf{b}_j$ of $\mathbf{L}N_2$ samples. Every block $\mathbf{b}_j$ has an overlap of $\mathbf{L}N_2 - 5$ with the previous one, that is, the difference between the blocks $\mathbf{b}_j$ and $\mathbf{b}_{j+1}$ is only 5 samples shifted to the right. From each block $\mathbf{b}_j$, N_2 bits are retrieved using the extraction algorithm of Section 6.1. Those bits represent a potential codeword $\tilde{C}^j$. Also, for each block $\mathbf{b}_j$ the log-likelihood ratio is computed with:

$$\mathbf{llr}_j = \frac{2\mathbf{r}_j \mu}{\sigma^2}, \tag{6.3}$$

where $\mathbf{r}_j = A' - B'$ is a vector with soft-information from the block $\mathbf{b}_j$, σ^2 is the noise variance and μ is the mean of $|A - B|$.

Each $\mathbf{llr}_j$ is decoded with only one iteration of SPA, thus, an updated codeword $\hat{C}^j$ and its syndrome $\mathcal{B}^j$ are obtained. The metric $\mathcal{M}_j = d_H(\tilde{C}^j, \hat{C}^j) + \mathcal{B}^j$ is used to decide whether a codeword $\tilde{C}^j$ is synchronized, where $d_H(\tilde{C}^j, \hat{C}^j)$ is the hamming distance between $\tilde{C}^j$ and $\hat{C}^j$. Therefore the codeword $\tilde{C}^j$ that minimize the metric $\mathcal{M}_j$ is selected as the synchronized codeword $\tilde{C}^{syn}$, and it is decoded using full-iterations of SPA.

6.3 Results and comparisons

The watermark was embedded in random segments of 30 seconds taken from three different audio files: *Egmont Op. 84* (7 min.), *Billie Jean* by M. Jackson (4 min.) and *A change of seasons* by Dream Theater (21 min.). Those audio files are mono in WAVE format, sampled at 44.1 kHz/16 bits. The LDPC code is a random half-rate code with code length 96.

The parameters that control the embedding phase were set to the same value used in [87]. $L_1 = L_2 = L_3 = 340$, γ was initially set to be 0.05 and the audio quality was controlled with

the psychoacoustic model in silence using a quality of 85%.

Table 6.1 shows the BER results for Lie's method and for our algorithm after MP3 and low-pass filtering. In general, Lie's method is robust against several audio manipulations however with strong attacks, e.g. MP3 at 64 kbps, the BER is not low enough. Our contribution to Lie's method produces more robust watermarks that do not need synchronization codes. The improvement in robustness might be small but the payload is increased as well. In [87] is not mentioned the length of the convolutional code N_2, therefore is difficult to quantify the payload benefit. However, in Fig. 6.2 is clear that our method has a benefit of 20 bits per each block of embedded information.

The results of our method, see Table 6.1, were a combination of two attacks: MP3 or low-pass filtering with clipping attack. The number of continuous clipped samples was chosen randomly in each iteration from an uniform distribution between $[1, \mathbf{L}N_2]$. In Lie's method, we assumed that the system was in perfect synchrony and only attacks like compression or low-pass filtering were implemented.

Table 6.1: BER results against MP3 compression and low-pass filtering.

Attacks		**Lie's method [87]** (BER)		**Proposed method** (BER)
		uncoded	*coded*	*coded*
MP3	160 kbps	0.0006	0.0000	0.0000
	128 kbps	0.0104	0.0001	0.0000
	64 kbps	0.0945	0.0259	0.0000
Low-pass	10 kHz	0.0000	0.0000	0.0000
	4 kHz	0.0280	0.0007	0.0000

It is expected that certain audio segments have stronger watermarks because the watermark's strength depends on the psychoacoustic model. Therefore, the watermark reliability varies depending on the audio file. That explains why different BER can be obtained with the same method but different audio files. In our comparison both methods, Lie's and ours, were tested with the same audio files.

Due to the good error correction capability of LDPC codes is not necessary to search the watermark in every audio sample. Searching the watermark every 5 samples is enough to decode the correct information, this explains the length of the overlap in Section 6.2.2. Clearly, if the search is done every 5 samples instead of every sample, the complexity is reduced. However, perfect alignment cannot be produced. In our opinion, perfect alignment is not important if the embedded watermark is retrieved correctly.

LDPC codes increase its performance with larger code length. Therefore, if larger LDPC codes were used the robustness would be increased. Small codes were used in this proposal because computational complexity restrictions. However, if smarter searches were implemented

then better LDPC codes could be used.

6.4 Conclusion

In this Chapter we proposed a modification to Lie's audio watermark algorithm in time-domain. The modification produces a more robust algorithm because LDPC codes are used instead of convolutional codes. The payload is also increased because synchronization codes are avoided but the synchrony is still kept, even after combination of attacks.

Our self-synchronous decoding algorithm is based on the good CP-characteristics of LDPC codes. The algorithm is not capable to produce prefect alignment, however the watermark is perfectly recovered. This misalignment is less than 10 audio samples, the same as in Lie's algorithm.

The synchronized codeword is searched in audio segments with size depending on the code length. Therefore small LDPC codes were used in order to keep the computational complexity low. However if the code length is increased our proposal becomes more robust. As future work, we are planning to implement smart searches which allow us to use larger codes with acceptable complexity.

7

Concluding remarks

We have studied watermarking algorithms for audio files from the communications point of view. The major results were two new algorithms: the first is an audio watermarking algorithm which achieves the highest embedding capacity robust to MP3 compression at 64 kbps and the second is an algorithm to recovering synchronization with LDPC codes that can be applied to general transmissions.

Quantization index modulation (QIM) algorithms were our main concern because they are embedding techniques that can achieve very high payloads. At first, a general watermarking scheme on wavelet domain was proposed and tested with different QIM-based techniques like dither modulation (DM) and spread-transform dither modulation (ST).

ST showed better performance against Gaussian noise achieving a robustness of watermark-to-noise ratio of -25 dB. However DM had better performance against common signal processing like compression and low-pass filtering. Thus, in further watermark algorithms, the watermark was embedded with DM in certain frequencies which were found to be robust to MP3 compression.

The weakness of QIM-based algorithms is well-known but using a concatenation of low-density parity-check (LDPC) codes and repetition codes the robustness was increased well enough to endure highly lossy compressions. Our analysis included many kind of LDPC codes from which a half-rate Margulis code had the best performance and yet its code length of 2640 is acceptable to watermarking purposes.

Decoding of LDPC codes means knowledge of the statistics of the channel noise. We proposed an accurate fashion to compute the statics of the channel noise due to MP3 compression. This idea allows us to develop two variations of the main watermarking algorithm: blind and semi-blind decoding. With the former we implemented a basic idea using pilot symbols to compute the channel noise, and the algorithm performs very similar, achieving 193.8 bps with BER = .08, to the best algorithm from the literature proposed by Bhat. The latter achieves 8.2 bps better than Bhat's algorithm with a BER $< 10^{-5}$.

The embedding capacity was over performed by defining erasures in likely noisy segments of the audio. Thus, the LDPC decoder improved its performance. The final embedding capacity achieved was of 229.7 bps which has 33.7 bps of benefit over Bhat's proposal.

We proposed a new self-synchronous algorithm with LDPC codes which recovers the synchronization without the aid of any extra-bits, assuming that the desynchronization was produced by a continuous deletion in the channel. Clipping attack produces desynchronization in audio watermarking because it clips samples from the audio. The algorithm to recovering the synchronization was thought with the clipping attack on mind. However the algorithm was explained for general transmissions due to it can be applied to any coded transmission with LDPC codes. The algorithm achieves synchronization with several codes but cyclic codes. With a very small LDPC code, the performance its 3 dB away from synchronized transmission; however the synchronization is achieved without any pilot symbols. The larger the length of the code the closer our algorithm performs from synchronized transmission. For example with a half-rate LDPC code of length 504, the performance of our algorithm is the same than synchronized transmission for $E_b/N_0 = 4$ dB.

The self-synchronized algorithm achieves synchronization with or without redundant codewords always that the code has good cyclically permutable characteristics. We also introduced an idea about how to bound the errors in the algorithm, this idea was illustrated by the maximum likelihood decoding of a small code.

Finally we combine the synchronous algorithm with watermarking. The result is a watermark algorithm with higher robustness and payload. The algorithm was developed with an idea proposed by Lie *et ál.* for time-domain watermarking. We modified the synchronization process and the coded-watermark in Lie's algorithm. Our algorithm uses LDPC codes to encode the watermark and the synchronization codes in Lie's algorithm were changed to our self-synchronous algorithm. The experimental results show that our algorithm is capable to recover synchronization after combination of attacks, e.g. MP3 compression and clipping attack.

7.1 Reflections about future research

Digital audio watermarking was conceived as potential solution to protect the copyrights of digital works. However due to copyright protection is a very demanding and coarse task there are still many open problems to achieve a reliable system. However, watermarking is used in many other applications nowadays.

In summary, the next topics might be considered as future work.

- A blind algorithm was introduced in Section 4.3.2. The algorithm computes the statistics of the channel noise using a basic technique with pilot symbols. Improvements can

be applied if a better channel characterization algorithm was used.

- In Section 4.4 we introduce a technique which defines erasures based on a correlation between the noise and the characteristics of the audio file. The technique can be improved searching a better correlation and also can be extended to other attacks like additive noise, echo addition, etc.

- Related with the self-synchronizable algorithm of Chapter 5. Considering that cyclically permutable (CP) codes have disadvantage of low-rate and that non-CP codes generates *mis-decoding* in our algorithm, there is an interesting question about whether a CP-code has better trade-offs of performance-rate than a non-CP code with good CP-characteristics.

- We combine the self-synchronous algorithm with time-domain watermarking in Chapter 6. However due to complexity restrictions we used small LDPC codes, length lower than 100. Smart searches are necessary in order to exploit the error correction capabilities of large LDPC codes.

- Our last proposal includes synchronization for watermark schemes on time. However, synchronization with algorithms in frequency is still very computational costly.

List of contributions

Related to this dissertation

Journals

1. **R. Martínez-Noriega**, K. Yamaguchi and K. Kobayashi, "Self-synchronizable decoding algorithms for transmission with redundant information at decoder," *IEICE Trans. on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E93-A, no. 11, Nov. 2010. (The contents of Chapter 5)

2. **R. Martínez-Noriega**, M. Nakano, B. Kurkoski and K. Yamaguchi, "High payload audio watermarking: toward channel characterization of MP3 compression," *Journal of Information Hiding and Multimedia Signal Processing (JIHMSP)*, vol. 2, no. 2, pp. 92-108, April 2011. (To appear. The contents of Chapter 4)

International conferences

1. **R. Martínez-Noriega**, B. Kurkoski, K. Yamaguchi and K. Kobayashi, "Recovering synchronization with iterative decoders: LDPC codes," *Int. Symposium on Information Theory and its Applications (ISITA)*, Taiwan, October 2010. (The contents of Chapter 5)

2. **R. Martínez-Noriega**, M. Nakano, and Yamaguchi, "Self-synchronous time-domain audio watermarking based on coded-watermarks," *Int. Symposium on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, Germany, October 2010. (The contents of Chapter 6)

3. **R. Martínez-Noriega**, M. Nakano and K. Yamaguchi, "On the channel characteristic of dither modulation data hiding for MP3 compression," *IEEE Int. Conf. on Intelligent Information Hiding and Multimedia Signal Processing*, Kyoto Japan, September 2009. (The contents of Chapter 4)

4. **R. Martínez-Noriega**, M. Nakano, B. Kurkoski, K. Yamaguchi and K. Kobayashi, "Analysis of QIM-based audio watermarking using LDPC codes," *IEEE Midwest Symposium on Circuits and Systems (MWSCAS)*, Cancun Mexico, August 2009. (The contents of Chapter 3)

5. **R. Martínez-Noriega**, M. Nakano and K. Yamaguchi, "Spread-Transform dither modulation for audio watermarking and its improvements with LDPC codes," *Triangle Symposium on Advanced ICT (TriSAI)*, Japan, October 2009. (The contents of Chapter 3)

6. **R. Martínez-Noriega**, M. Nakano, B. Kurkoski, K. Yamaguchi and K. Kobayashi, "Robustness analysis of audio QIM methods with LDPC codes," *Triangle Symposium on Advanced ICT (TriSAI)*, Korea, October 2008. (The contents of Chapter 3)

Domestic conferences

1. **R. Martínez-Noriega**, I. Abe, K. Yamaguchi, "Study on cyclically permutable property on error correcting codes", *IEICE Technical Report*, IT2009-55, pp. 23-28, January 2010. (The contents of Chapter 5)

2. **R. Martínez-Noriega**, M. Nakano, B. Kurkoski and K. Yamaguchi "Decoding strategies based on erasures for audio watermarking with LDPC codes," *Symposium on Information Theory and its Applications (SITA)*, Yamaguchi Japan, December 2009. (The contents of Chapter 4)

3. **R. Martínez-Noriega**, M. Nakano, B. Kurkoski, K. Yamaguchi and K. Kobayashi, "Adaptive dither modulation audio watermarking with LDPC decoding based on Rayleigh fading channel," *Symposium on Cryptography and Information Security (SCIS)*, Otsu Japan, January 2009. (The contents of Chapter 3)

4. **R. Martínez-Noriega**, M. Nakano, B. Kurkoski, K. Yamaguchi and K. Kobayashi, "QIM audio watermarking systems using LDPC code with soft decoding," *Symposium on Cryptography and Information Security (SCIS)*, Miyazaki Japan, January 2008. (The contents of Chapter 3)

5. **R. Martínez-Noriega**, H. Kang, B. Kurkoski, K. Yamaguchi and M. Nakano, "QIM Audio watermarking with distortion compensated property," *Symposium on Cryptography and Information Security (SCIS)*, Sasebo Japan, January 2007. (The contents of Chapter 3)

6. I. Abe, **R. Martínez-Noriega**, M. Nakano, B. Kurkoski, K. Yamaguchi and K. Kobayashi "Performance of quantization-based watermarking methods using LDPC codes," *Symposium on Information Theory and its Applications (SITA)*, Kinugawa Japan, October 2008. (The contents of Chapter 3)

Others

International conferences

1. **R. Martínez-Noriega**, R. Reyes, C. Cruz, M. Nakano and H. Perez, "A DWT-based video watermarking scheme resilient to MPEG-2 compression and collusion attacks," *Int. Symposium on Information Theory and its Applications (ISITA)*, New Zealand, December 2008.

2. **R. Martínez-Noriega**, H. Kang, B. Kurkoski, K. Yamaguchi and M. Nakano, "Audio watermarking based on wavelet transform and quantization index modulation", *The 22nd International Technical Conference on Circuits/Systems, Computers and Communications 2007 (ITC-CSCC)*, Busan Korea, June 2007.

3. **R. Martínez-Noriega**, H. Kang, B. Kurkoski, K. Yamaguchi, K. Kobayashi and M. Nakano, "Increasing robustness of audio watermarking DM using ATHC codes", *IEEE Mexican Conference on Information Security*, Mexico, November 2006.

4. C. Velasco, M. Nakano, H. Perez, **R. Martínez-Noriega** and K. Yamaguchi, "Adaptive JPEG steganography using convolutional codes and synchronization bits in DCT domain," *IEEE Midwest Symposium on Circuits and Systems (MWSCAS)*, Cancun Mexico, August 2009.

5. J. Lopez-Hernandez, **R. Martínez-Noriega**, M. Nakano and K. Yamaguchi, "Detection of BPCS-steganography using SMWFC steganalysis and SVM," *Int. Symposium on Information Theory and its Applications (ISITA)*, New Zealand, December 2008.

6. C. Velasco, **R. Martínez-Noriega**, K. Yamaguchi and M. Nakano, "Data hiding in the DCT domain using entropy thresholding scheme with a mapping matrix," *Triangle Symposium on Advanced ICT (TriSAI)*, Korea, October 2008.

Domestic conferences

1. Y. Iriguchi, **R. Martínez-Noriega** and K. Yamaguchi, "A new scheme for a collusion-secure code to improve its trace back rate in content distribution systems," *Symposium on Cryptography and Information Security (SCIS)*, Otsu Japan, January 2009.

2. C. Velasco, **R. Martínez-Noriega**, M. Nakano and K. Yamaguchi, "Adaptive image data hiding using DCT steganographic and error control codes," *Symposium on Cryptography and Information Security (SCIS)*, Otsu Japan, January 2009.

3. Y. Kobayashi, I. Abe, **R. Martínez-Noriega** and K. Yamaguchi "Audio watermarking based on LDPC code for clipping attack," *Symposium on Cryptography and Information Security (SCIS)*, Otsu Japan, January 2009.

4. Y. Kobayashi, I. Abe, **R. Martínez-Noriega** and K. Yamaguchi, "Audio watermarking based on LDPC codes for clipping attack and the decoding-extracting algorithm with self-synchronization," *Symposium on Information Theory and its Applications (SITA)*, Kinugawa Japan, October 2008.

5. C. Velasco, M. Nakano, **R. Martínez-Noriega** and K. Yamaguchi, "Steganography in color images using entropy thresholding scheme," *Symposium on Information Theory and its Applications (SITA)*, Kinugawa Japan, October 2008.

6. J. C. Lopez-Hernandez, **R. Martínez-Noriega**, M. Nakano and K. Yamaguchi, "A survey of wavelet domain steganalysis methods," *Symposium on Cryptography and Information Security (SCIS)*, Miyazaki Japan, January 2008.

7. J. Kubota, **R. Martínez-Noriega**, H. Kang, B. Kurkoski and K. Yamaguchi, "A proposal for an AM-based watermarking system using an attack-adapted decoder," *Symposium on Cryptography and Information Security (SCIS)*, Sasebo Japan, January 2007.

References

[1] http://www.apple.com/pr/library/2010/02/25itunes.html.

[2] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, *Digital Watermarking and Steganography*, 2nd ed. Burlington, USA: Morgan Kaufmann Publishers, 2008.

[3] R. H. Morelos-Zaragoza, *The Art of Error Correcting Coding*. John Wiley & Sons, 2006.

[4] R. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, vol. 8, no. 10, pp. 21–28, 1962.

[5] R. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533 – 547, sep 1981.

[6] I. Cox, J. Kilian, F. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions on Image Processing*, vol. 6, no. 12, pp. 1673 –1687, Dec. 1997.

[7] I. Cox, J. Kilian, T. Leighton, and T. Shamoon, "A secure, robust watermark for multimedia," in *Information Hiding*, ser. Lecture Notes in Computer Science, R. Anderson, Ed. Springer Berlin/Heidelberg, 1996, vol. 1174, pp. 185–206.

[8] J. R. Smith and B. O. Comiskey, "Modulation and information hiding in images," in *Proceedings of the First International Workshop on Information Hiding*. London, UK: Springer-Verlag, 1996, pp. 207–226.

[9] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, "Techniques for data hiding," *IBM Systems Journal*, vol. 35, no. 3-4, pp. 313–336, 1996.

[10] L. Boney, A. Tewfik, and K. Hamdy, "Digital watermarks for audio signals," in *Proceedings of the Third IEEE International Conference on Multimedia Computing and Systems*, 17-23 1996, pp. 473 –480.

[11] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes. 1," in *IEEE International Conference on Communications (ICC)*, vol. 2, May 1993, pp. 1064 –1070 vol.2.

[12] D. MacKay and R. Neal, "Near shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 32, no. 18, p. 1645, 29 1996.

[13] D. MacKay, "Good error-correcting codes based on very sparse matrices," in *IEEE International Symposium on Information Theory*, 29 1997, p. 113.

[14] I. Cox, M. Miller, and A. McKellips, "Watermarking as communication with side information," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1127–1141, 1999.

[15] J. Chou, S. Sandeep Pradhan, and K. Ramchandran, "On the duality between distributed source coding and data hiding," in *Conference Record of the Thirty-Third Asilomar Conference on Signals, Systems, and Computers*, vol. 2, 1999, pp. 1503 –1507 vol.2.

[16] B. Chen and G. Wornell, "An information-theoretic approach to the design of robust digital watermarking systems," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 4, Mar. 1999, pp. 2061 –2064 vol.4.

[17] C. E. Shannon, "Channels with side information at the transmitter," *IBM J. Res. Dev.*, vol. 2, pp. 289–293, October 1958.

[18] M. Miller, I. Cox, and J. Bloom, "Informed embedding: exploiting image and detector information during watermark insertion," in *International Conference on Image Processing*, vol. 3, 2000, pp. 1 –4 vol.3.

[19] J. Chou, S. Pradhan, L. El Ghaoui, and K. Ramchandran, "Watermarking based on duality with distributed source coding and robust optimization principles," in *International Conference on Image Processing*, vol. 1, 2000, pp. 585 –588 vol.1.

[20] P. Moulin and J. O'Sullivan, "Information-theoretic analysis of information hiding," *IEEE Transactions on Information Theory*, vol. 49, no. 3, pp. 563 – 593, Mar. 2003.

[21] B. Chen and G. Wornell, "Quantization index modulation: A class of provably good methods for digital watermarking and information embedding," *IEEE Trans. on Inform. Theory*, vol. 47, no. 4, 2001.

[22] M. Costa, "Writing on dirty paper," *IEEE Trans. on Inform. Theory*, vol. IT-29, pp. 731–739, 1983.

[23] F. Perez-Gonzalez, C. Mosquera, M. Barni, and A. Abrardo, "Rational dither modulation: A high-rate data-hiding method invariant to gain attacks," *IEEE Trans. on Sig. Proc.*, vol. 53, no. 10, pp. 3960–3975, 2005.

[24] S. Baudry, P. Nguyen, and H. Maitre, "Channel coding in video watermarking: use of soft decoding to improve the watermark retrieval," in *International Conference on Image Processing*, vol. 3, 2000, pp. 25 –28 vol.3.

[25] N. Abdulaziz and K. Pang, "Robust data hiding for images," in *International Conference on Communication Technology Proceedings (WCC-ICCT)*, vol. 1, 2000, pp. 380 –383 vol.1.

[26] S. Baudry, J. F. Delaigle, B. Sankur, B. Macq, and H. Matre, "Analyses of error correction strategies for typical communication channels in watermarking," *Signal Processing*, vol. 81, no. 6, pp. 1239–1250, 2001.

[27] 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," *Signal Processing*, vol. 81, pp. 1215–1238, June 2001.

[28] C. Desset, B. Macq, and L. Vandendorpe, "Block error-correcting codes for systems with a very high BER: Theoretical analysis and application to the protection of watermarks," *Signal Processing. Image Comms.*, vol. 17, no. 5, pp. 409–421, 2002.

[29] L. Gu, Y. Fang, and J. Huang, "Revaluation of error correcting coding in watermarking channel*," *Lecture Notes in Computer Science*, vol. 3810, pp. 274–287, 2005.

[30] D. Kirovski and H. Malvar, "Spread-spectrum watermarking of audio signals," *IEEE Transactions on Signal Processing*, vol. 51, no. 4, pp. 1020 – 1033, apr 2003.

[31] N. Cvejic, D. Tujkovic, and T. Seppanen, "Increasing robustness of an audio watermark using turbo codes," in *Int. Conf. on Multimedia and Expo*, 2003, pp. 217–220.

[32] A. Bastug and B. Sankur, "Improving the payload of watermarking channels via ldpc coding," *IEEE Signal Processing Letters*, vol. 11, no. 2, pp. 90 – 92, Feb. 2004.

[33] F. Balado, F. Pérez-González, and P. Comesana, "Blind iterative decoding of side-informed data hiding using the expectation-maximization algorithm," *Security, Steganography, and Watermarking of Multimedia Contents VI*, vol. 1, pp. 805–815, June 2004.

[34] N. Cvejic and T. Seppänen, "Spread spectrum audio watermarking using frequency hopping and attack characterization," *Signal Process.*, vol. 84, no. 1, pp. 207–213, 2004.

[35] J. J. Eggers and B. Girod, "Quantization effects on digital watermarks," *Signal Processing*, vol. 81, pp. 239–263, February 2001.

[36] F. Balado, K. Whelan, G. Silvestre, and N. Hurley, "Joint iterative decoding and estimation for side-informed data hiding," *IEEE Transactions on Signal Processing*, vol. 53, no. 10, pp. 4006 – 4019, Oct. 2005.

[37] D. Coumou and G. Sharma, "Watermark synchronization for feature-based embedding: Application to speech," in *IEEE International Conference on Multimedia and Expo*, july 2006, pp. 849 –852.

[38] C. Dikici, K. Idrissi, and A. Baskurt, "Dirty-paper writing based on ldpc codes for data hiding," in *Multimedia Content Representation, Classification and Security*, ser. Lecture Notes in Computer Science. Springer Berlin/Heidelberg, 2006, vol. 4105, pp. 114–120.

[39] J. Hernandez, M. Nakano, and H. Meana, "Rational dither modulation in audio signals," in *50th Midwest Symp. Circuits and Systems*, 2007, pp. 109–112.

[40] A. Deshpande and K. Prabhu, "A substitution-by-interpolation algorithm for watermarking audio," *Signal Processing*, vol. 89, no. 2, pp. 218 – 225, 2009.

[41] V. Bhat, I. Sengupta, and A. Das, "An audio watermarking scheme using singular value decomposition and dither-modulation quantization," *Multimedia Tools and Applications*, april 2010.

[42] W. M. Tomberlin, L. G. Mackensie, and P. K. Bennett, "System for transmitting and receiving coded entertainment programs," U.S. Patent 2,630,525, 1953.

[43] W. E. Noller, "Inband signalling system," U.S. Patent 3,061,783, 1962.

[44] R. H. Baer, "Digital video modulation and demodulation system," U.S. Patent 3,993,861, 1976.

[45] R. S. Broughton and W. C. Laumeister, "Interactive video method and apparatus," U.S. Patent 4,807,031, 1989.

[46] R. Dolby, "Apparatus and method for the identification of specially encoded fm stereophonic broadcasts," U.S. Patent 4,281,217, 1981.

[47] N. Aoki, "Improvement of a band extension technique for G.711 telephony speech by using steganography," in *5th Int. Conf. on Intelligent Information Hiding and Multimedia Signal Processing*, 2009, pp. 487–490.

[48] N. Cvejic, N. Cvejic, and T. Seppanen, *Digital Audio Watermarking Techniques and Technologies: Applications and Benchmarks*. IGI Global, 2007.

[49] D. Pan, "A tutorial on MPEG/audio compression," *IEEE Multimedia*, vol. 2, no. 2, pp. 60 –74, summer 1995.

[50] S. Andreas, P. Ted, and V. Atti, *Audio Signal Processing and Coding*. John Wiley & Sons, 2007.

[51] S. Lin and D. J. Costello, *Error Control Coding, Second Edition*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2004.

[52] S. Katzenbeisser and F. A. Petitcolas, Eds., *Information Hiding Techniques for Steganography and Digital Watermarking*. Norwood, MA, USA: Artech House, Inc., 2000.

[53] J. Eggers, R. Bauml, R. Tzschoppe, and B. Girod, "Scalar costa scheme for information embedding," *IEEE Transactions on Signal Processing*, vol. 51, no. 4, pp. 1003 – 1019, apr 2003.

[54] M. Steinebach, F. Petitcolas, F. Raynal, J. Dittmann, C. Fontaine, S. Seibel, N. Fates, and L. Ferri, "Stirmark benchmark: audio watermarking attacks," in *International Conference on Information Technology: Coding and Computing*, apr 2001, pp. 49 –54.

[55] http://wwwiti.cs.uni-magdeburg.de/~alang/smba.php.

[56] Q. Li and I. Cox, "Using perceptual models to improve fidelity and provide resistance to valumetric scaling for quantization index modulation watermarking," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 2, pp. 127 –139, june 2007.

[57] X. Li and H. Yu, "Transparent and robust audio data hiding in subband domain," in *International Conference on Information Technology: Coding and Computing*, 2000, pp. 74 –79.

[58] I.-K. Yeo and H. J. Kim, "Modified patchwork algorithm: a novel audio watermarking scheme," *IEEE Transactions on Speech and Audio Processing*, vol. 11, no. 4, pp. 381 – 386, july 2003.

[59] W. Rangding, X. Dawen, C. Jiner, and D. Chengtou, "Digital audio watermarking algorithm based on linear predictive coding in wavelet domain," in *International Conference on Signal Processing*, vol. 3, 31 2004, pp. 2393 – 2396 vol.3.

[60] C.-Y. Chang, W.-C. Shen, and H.-J. Wang, "Using counter-propagation neural network for robust digital audio watermarking in dwt domain," in *IEEE International Conference on Systems, Man and Cybernetics*, vol. 2, 8-11 2006, pp. 1214 –1219.

[61] W. Li, X. Xue, and P. Lu, "Localized audio watermarking technique robust against time-scale modification," *IEEE Transactions on Multimedia*, vol. 8, no. 1, pp. 60 – 69, feb 2006.

[62] S. Xiang and J. Huang, "Histogram-based audio watermarking against time-scale modification and cropping attacks," *IEEE Transactions on Multimedia*, vol. 9, no. 7, pp. 1357 –1372, nov. 2007.

[63] S. Xiang, H. J. Kim, and J. Huang, "Audio watermarking robust against time-scale modification and mp3 compression," *Signal Process.*, vol. 88, no. 10, pp. 2372–2387, 2008.

[64] E. Erçelebi and L. Batakçı, "Audio watermarking scheme based on embedding strategy in low frequency components with a binary image," *Digit. Signal Process.*, vol. 19, no. 2, pp. 265–277, 2009.

[65] M. Fan and H. Wang, "Chaos-based discrete fractional sine transform domain audio watermarking scheme," *Computers & Electrical Engineering*, vol. 35, no. 3, pp. 506 – 516, 2009.

[66] X.-Y. Wang, P.-P. Niu, and H.-Y. Yang, "A robust digital audio watermarking based on statistics characteristics," *Pattern Recognition*, vol. 42, no. 11, pp. 3057 – 3064, 2009.

[67] D. Megías, J. Serra-Ruiz, and M. Fallahpour, "Efficient self-synchronised blind audio watermarking system based on time domain and FFT amplitude modification," *Signal Processing*, vol. In Press, Corrected Proof, pp. –, 2010.

[68] S. Wu, J. Huang, D. Huang, and Y. Shi, "Efficiently self-synchronized audio watermarking for assured audio data transmission," *IEEE Transactions on Broadcasting*, vol. 51, no. 1, pp. 69 – 76, march 2005.

[69] J. J. Garcia-Hernandez, M. Nakano-Miyatake, and H. Perez-Meana, "Data hiding in audio signal using rational dither modulation," *IEICE Electronics Express*, vol. 5, no. 7, pp. 217–222, 2008.

[70] M. Fallahpour and D. Megias, "High capacity audio watermarking using FFT amplitude interpolation," *IEICE Electronics Express*, vol. 6, no. 14, pp. 1057–1063, 2009.

[71] M. Fallahpour and D. Megías, "High capacity audio watermarking using the high frequency band of the wavelet domain," *Multimedia tools and Applications*, 2010.

[72] M. Arnold, M. Schmucker, and S. D. Wolthusen, *Techniques and Applications of Digital Watermarking and Content Protection.* Norwood, MA, USA: Artech House, Inc., 2003.

[73] T. Thiede, W. C. Treurniet, R. Bitto, C. Schmidmer, T. Sporer, J. G. Beerends, and C. Colomes, "PEAQ - the itu standard for objective measurement of perceived audio quality," *J. Audio Eng. Soc*, vol. 48, no. 1/2, pp. 3–29, 2000.

[74] S. W. Golomb and L. R. Welch, "Comma-free codes," *Canadian Journal of Mathematics*, vol. 10, no. 2, pp. 202–209, 1958.

[75] H. Imai, "A construction method for path-invariant comma-free codes," *IEEE Trans. on Inf. Theory*, vol. 20, no. 4, pp. 555–559, 1974.

[76] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Cybernetics and Control Theory*, vol. 10, no. 8, pp. 707–710, 1966.

[77] E. Gilbert, "Cyclically permutable error-correcting codes," *IEEE Transactions on Information Theory*, vol. 9, no. 3, pp. 175 – 182, jul. 1963.

[78] M. Kuribayashi and M. Morii, "Spread spectrum watermark with self-synchronization capability," in *Proceedings of Symposium of Cryptography and Information Security (SCIS)*, Sasebo, Japan, Jan. 2007.

[79] M. Kuribayashi and H. Tanaka, "How to generate cyclically permutable codes from cyclic codes," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4660 –4663, oct. 2006.

[80] W. Matsumoto and H. Imai, "Blind synchronization with enhanced sum-product algorithm for low-density parity-check codes," in *Proc. of The 5th Int. Symp. on Wireless Personal Mult. Comms.*, Oct. 2002, pp. 966–970.

[81] H. Steendam, N. Noels, and M. Moeneclaey, "Iterative carrier phase synchronization for low-density parity-check coded systems," in *IEEE International Conference on Communications, ICC*, vol. 5, 11-15 2003, pp. 3120 – 3124 vol.5.

[82] R. Martínez-Noriega, M. Nakano, and K. Yamaguchi, "Self-synchronous time-domain audio watermarking based on coded-watermarks," in *Proceedings of International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Oct. 2010.

[83] R. Martinez-Noriega, I. Abe, and K. Yamaguchi, "Self-synchronizable decoding algorithms for transmission with redundant information at decoder," *IEICE Trans. on Fundamentals of Electronics Comms. and Computer Science*, 2010.

[84] R. Martínez-Noriega, K. Yamaguchi, and K. Kobayashi, "Recovering synchronization with iterative decoders: LDPC codes," *Proc. of Int. Symp. on Inf. Theory and its Apps.*, 2010.

[85] J.-S. Pan, H.-C. Huang, L. C. Jain, and W.-C. Fang, *Intelligent Multimedia Data Hiding: New Directions*. Springer Publishing Company, Incorporated, 2007.

[86] K. Yamamoto and M. Iwakiri, "Real-time audio watermarking based on characteristics of PCM in digital instrument," *Information Hiding and Multimedia Signal Processing (JIH-MSP)*, vol. 1, no. 2, pp. 59–71, 2010.

[87] W. N. Lie and L. C. Chang, "Robust and high-quality time-domain audio watermarking based on low-frequency amplitude modificacion," *IEEE Trans. on Multimedia*, vol. 8, no. 1, pp. 46–59, 2006.

Acknowledgements

I am thankful to my advisor, Prof. Yamaguchi Kazuhiko for the guidance and advices he has given me through these years not only in the academia but the life itself. I am also grateful to Prof. Kurkoski Brian who gave me a lot of his time in fruitful discussion which opened my mind to good research directions.

I would like to thank to all my lab mates and friends who have made my life very comfortable in Japan. Thanks to Dr. Kang Hyunho who help me a lot when I just arrived to Japan. I am grateful to all the staff of the exchange program, JUSST, at the University of Electro-Communications, especially with Prof. Suzuki Masahisa who has been kind to me. Thank you to my formal advisor in Mexico, Prof. Nakano Mariko who started the engine of this journey and encourage me to achieve higher goals.

Muchas gracias a toda mi familia que siempre ha estado al pendiente de mí. Gracias a mi abuelo Enrique donde quiera que se encuentre, por sus palabras de ánimo y valentía. Gracias a Alexis por su paciencia, quien ha estado todo este tiempo brindándome su apoyo. Muchas gracias a mi hermano Edgar quien ha sido una fuente de inspiración, amistad y apoyo. Estoy muy agradecido y satisfecho con mis padres Raúl y Edith por todo lo que me han enseñado a lo largo de mi vida, su gran amor y soporte.

Finally, I would like to dedicate this dissertation to my mother Edith, the strongest woman on the Earth. Congratulations for your recovering!

Thank you so much!

Raúl.

Author biography

Raúl Martínez-Noriega was born in Mexico City, Mexico, on September 1982. He received the B.Sc. and M.Sc. degrees from the National Polytechnic Institute (IPN from the name in Spanish - Instituto Politecnico Nacional) of Mexico in 2004 and 2007 respectively. He has been with the Department of Information and Communications Engineering of The University of Electro-Communications in Tokyo, Japan, towards the Doctoral degree.

He was exchange student in UEC from 2006 to 2007 where he received the best exchange research student award.

His research interest are coding theory especially LDPC codes, information theoretic security towards development of secure coding for the wire-tap channel, and watermarking in audio and images.