uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

u Ottawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Bingwei Chen

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M. (Computer Science)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Adaptive Watermarking Algorithms for MP3 Compressed Audio Signals

TITRE DE LA THÈSE / TITLE OF THESIS

Jiying Zhao

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Abdulmotaleb El Saddik

Peter X. Liu

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Adaptive Watermarking Algorithms for MP3 Compressed Audio Signals

by

Bingwei Chen

A thesis submitted to
the Faculty of Graduate and Postgraduate Studies
in partial fulfillment of
the requirements for the degree of

Master of Computer Science

Ottawa-Carleton Institute for Computer Science
School of Information Technology and Engineering
University of Ottawa

Ottawa, Ontario, Canada

April 2008

Your file Votre référence
ISBN: 978-0-494-50862-6
Our file Notre référence
ISBN: 978-0-494-50862-6

Contents

List of Tables

List of Figures

List of Acronyms

BER	Bit Error Rate
CRC	Cyclic Redundancy Check
DFT	Discrete Fourier Transform
DSSS	Direct Sequence Spread Spectrum
FFT	Fast Fourier Transform
FSSS	Frequency Hopped Spread Spectrum
HAS	Human Auditory System
HVS	Human Visual System
IMDCT	Inverse Modified Discrete Cosine Transform
LSB	Least Significant Bit
MDCT	Modified Discrete Cosine Transform
MP3	MPEG-1 Audio Layer 3
MPEG	Moving Picture Experts Group
MPEG-2 AAC	MPEG-2 Advanced Audio Coding
PN-sequence	pseudo-random sequence
PSNR	Peak Signal-to-Noise Ratio
SNR	Signal to Noise Ratio
SPL	Sound Pressure Level

Acknowledgements

This thesis is dedicated to my parents, Xiaoyu Chen and Qixin Chen, who raised me, support me, love me and managed to give me exactly what I needed to get where I am today. I love you, Mom and Dad.

I am deeply grateful to my supervisor, Dr. Jiying Zhao. Without his encouragement and support I would never have finished.

Abstract

MPEG-1 Layer 3, known as MP3, has generated a significant popularity for distributing digital music over the Internet. MP3 compresses digital music with high ratio while keeping high sound quality. However, copyright issue is raised because of illegal copy, redistribution and various malicious attacks.

Digital watermarking is a technology that allows users to embed some imperceptible data into digital contents such as image, movie and audio data. Once a watermark is embedded into the original MP3 signal, it can be used to identify the copyright holder in order to prevent illegal copy and to verify the modification from the original content.

This thesis presents two novel adaptive watermarking algorithms for MP3 compressed audio signals for copyright protection. Based on Human Auditory System, the proposed algorithms calculate the energy of the original audio signal and apply Gaussian analysis on MP3 frames to adaptively adjust the watermarking coefficients. Watermark is embedded adaptively and transparently during the MP3 compression. The first watermarking algorithm detects watermark based on Gaussian distribution analysis. To enhance the security of the watermark, the second watermarking algorithm embeds random watermark pattern and uses correlation coefficient to detect watermark. Both algorithms support blind watermark detection and perform well. The first algorithm is more robust while the second algorithm is more secure.

LAME 3.96.2 open source was used as standard ISO MP3 encoder and decoder reference in this study. The experimental results show that the proposed watermarking algorithms can work on a variety of audio signals and survive most common signal manipulation and malicious attacks. As expected, the watermarking algorithms provide superior performance on MP3 compression.

Chapter 1

Introduction

1.1 Background

For the past few years, the MPEG-1 Layer 3 (MP3) audio compression standard has played a more and more important role in music industry and our life. Meanwhile, the illegal copy issue is raised. Therefore, copyright protection becomes a significant topic both in the academy and in the industry.

In the academy, there are several different techniques that can be used for information security, such as cryptography, steganography and watermarking. Each protection technology has its own specific goals and applications.

Cryptography is probably the most common method of protecting digital content. It is the encryption to convert data into a form called a cipher text that cannot be easily understood by unauthorized people. There are several types of cryptographic algorithms. Secret Key Cryptography (SKC) uses a single key for both encryption and decryption; Public Key Cryptography (PKC) uses one key for encryption and another for decryption; and Hash Functions uses a mathematical transformation to irreversibly

"encrypt" information. Cryptography is useful for transmission, but once decrypted, the content has no future protection [3].

Steganography is a term derived from Greek word steganos, which literally means 'covered writing', which is writing hidden messages in such a way that no one apart from the intended recipient knows of the existence of the message [4]. For example, in ancient Greece, people wrote messages on the wood, then covered it with wax so that it looked like an ordinary, unused tablet. The purpose of steganography is having a covert communication between two parties whose existence is unknown to a possible attacker. Steganography is in contrast to cryptography, where the existence of the message itself is not disguised, but the content is obscured.

Compared with those two information security methods, digital watermarking is the technique to embed watermark into host signal directly for the security protection. The watermark remains in the content in its original form and does not prevent a user from listening to, viewing, examining, or manipulating the content [5]. Unlike the idea of steganography, where the method of hiding the message may be secret and the message itself is secret, in watermarking, typically the watermark embedding process is known and the message does not have to be secret [5].

For copyright protection, nowadays, digital watermarking is one of the most widely deployed methods in industry. Komatsu and Tominaga have first used the term digital watermark in 1998. However, digital watermark was not introduced as a technique for copyright protection until the early 1990s [3]. The technique takes its name from paper watermarking on money as a security measure. It is a technique that allows an individual to add hidden copyright notices or other verification messages to various digital media, such as image, audio and video. Such hidden copyright notices, named watermarks, are a group of data stream embedded imperceptibly into the original signal.

In this thesis, we will research on digital audio watermarking algorithms especially on MPEG-1 Audio Layer 3 (MP3) compressed audio signals.

1.2 Objectives

MP3 is a very common and popular compression standard for audio signals. At the same time, MP3 is also a very severe attack against audio watermarking algorithms. Very few papers report digital watermarking algorithms that directly embed watermark in the MP3 domain and no paper has implemented adaptive watermarking algorithm for MP3 compressed audio signals. The primary objective of this thesis is to review what audio watermarking algorithms have been proposed in the literature, and to develop novel digital watermarking algorithms for MPEG-1, layer 3 (MP3) compressed audio signals. The proposed algorithms should have good performance against MP3 compression and other most common attacks, and should be secure. The watermark detection should not need the original audio signal.

1.3 Contributions

We proposed two novel audio watermarking algorithms in this thesis. The key feature in both algorithms is to embed watermark adaptively according to the original audio signal energy calculation and Human Auditory System. In the first algorithm, Gaussian distribution analysis is applied on MP3 frames during watermark embedding to calculate audio signal energy and to apply adaptive watermark coefficients according to the strength of the original audio signal. As a result, watermark is embedded adaptively and transparently into the original signal. This method allows embedding watermark

into various frequency regions without the traditional low-middle watermarking space limitation. Our experiments show this increases watermark space and improves the watermark reliability under MP3 compression.

In addition, the algorithm employs a dictionary to enhance the watermark security. Watermark owners can design their look-up table to represent one-bit watermark data. So the watermark bits are encrypted. Even though attackers can detect the embedded watermark binary sequence, they may be still unable to interpret its meaning, because only watermark owners know how to decode the watermark data.

To enhance watermark security, we proposed a correlation based watermarking algorithm that uses random watermark pattern during watermark embedding and correlation analysis for watermark detection. Watermark is extracted out according to the correlation coefficient threshold without need for original reference. Thus this algorithm uses blind watermark detection as well as adaptive watermark embedding.

These two watermarking algorithms have been tested using a variety of audio signals under MP3 compressions and common attacks such as low-pass filtering and Gaussian noise addition. Our test results show that the first algorithm gives better performance of robustness; while the second algorithm has enhanced security. The robustness of the second watermarking algorithm is sufficiently good, but is not as good as that of the first algorithm. Thus each algorithm can be used in different applications.

The following conference paper has been generated based on the first watermarking algorithm:

[1] **Bingwei Chen**, Jiying Zhao, and Dali Wang, An Adaptive Watermarking Algorithm for MP3 Compressed Audio Signals, I^2MTC 2008 - IEEE International Instrumentation and Measurement Technology Conference, Victoria, Vancouver Island, British Columbia, Canada, May 12-15, 2008.

1.4 Thesis Organization

The rest the thesis is organized as follows. In Chapter 2, an overview of MPEG Layer 3 Audio Compression (MP3) is presented. In Chapter 3, digital audio watermarking techniques with performance evaluation are given, and the existing literatures have been reviewed. In Chapter 4, our new adaptive audio watermarking algorithms and their implementations are presented. In Chapter 5, the experimental results and evaluation are given. In Chapter 6, conclusions are given and possible future work is pointed.

Chapter 2

MPEG Layer 3 Audio Compression (MP3)

2.1 Introduction to MPEG

MPEG stands for Moving Picture Experts Group, is the nickname given to a family of International Standards used for coding audio-visual information in a digital compressed format [1]. MPEG is originally the name given to the group of experts that developed these standards. The MPEG family of standards includes MPEG-1, MPEG-2 and MPEG-4, formally known as ISO/IEC-11172, ISO/IEC-13818 and ISO/IEC-14496.

2.1.1 MPEG Technology Overview

The major advantage of MPEG compared to other video and audio coding formats is that MPEG files are much smaller for the same quality. This is because MPEG uses very sophisticated compression techniques [1].

MPEG-1 video was originally designed with a goal of achieving acceptable video

quality at around 1.5 Mb/sec data rates and 352×240 pixel (29.97 frames per second) or 352×288 pixel (25 frames per second) resolution. While MPEG-1 applications are often of low resolutions and low bit rates, the standard allows any resolution less than 4095×4095. Nevertheless, most implementations were designed with the Constrained Parameter Bitstream specification in mind. Therefore, MPEG-1 video is used by the Video CD (VCD) format and less commonly by the DVD-Video format. The quality at standard VCD resolution and bit rate is near the quality and performance of a VHS tape. MPEG-1 audio has three layers. The most popular is MPEG-1 audio layer 3 known as MP3 [1]. One big disadvantage of MPEG-1 video is that it supports only progressive pictures. This deficiency helped prompt development of the more advanced MPEG-2 [1].

MPEG-2 describes a combination of lossy video compression and lossy audio compression (audio data compression) methods which permit storage and transmission of movies using currently available storage media and transmission bandwidth [1]. It is widely used as the format of digital television signals that are broadcast by terrestrial (over-the-air), cable, and direct broadcast satellite TV systems. MPEG-2 specifies that the raw frames be compressed into three kinds of frames: intra-coded frames (I-frames), predictive-coded frames (P-frames), and bidirectionally-predictive-coded frames (B-frames). The DVD standard uses MPEG-2 video [1]. MPEG-2 also introduces new audio encoding methods. These are low bitrate encoding with halved sampling rate, multichannel encoding with up to 5.1 channels and MPEG-2 AAC [1].

MPEG-4 is a standard used primarily to compress audio and visual (AV) digital data [1]. The key parts to be aware of MPEG-4 are MPEG-4 part 2 (MPEG-4 SP/ASP, used by codecs such as DivX, Xvid, Nero Digital and 3ivx and by Quicktime 6) and MPEG-4 part 10 (MPEG-4 AVC/H.264, used by the x264 codec, by Nero Digital AVC, by

Quicktime 7, and by next-gen DVD formats like HD DVD and Blu-ray Disc) [1].

2.1.2 Layers of MPEG-1 Audio Compression

Different layers of the coding system with increasing encoder complexity and performance can be used. For instance, an MPEG Audio Layer N decoder is able to decode bit stream data, which has been encoded in Layer N and all layers below N [1].

MPEG-1 Layer 1 has the lowest complexity. It uses polyphase filterbank to divide original audio signal into 32 equal frequency subbands with fixed segmentation to format the data into blocks. Psychoacoustic model I is used to determine the masking threshold and the adaptive bit allocation. Each block corresponds to 384 input PCM samples. Uniform quantization is used to quantize the audio samples. The theoretical minimum encoding and decoding delay for Layer I is about 19 ms [1]. The compression ratio for layer1 is 1:4 and the data rates for layer 1 is 384 kbps. Table 2.2 [6] shows the compression ratios and data rates for MPEG-1 layer1, 2, and 3.

MPEG-1 Layer 2 requires a more complex encoder and a slightly more complex decoder. Layer 2 also uses polyphase filterbank to divide original audio signal into 32 equal frequency subbands with fixed segmentation to format the data into blocks. Each block corresponds to 1152 PCM samples, which is calculated by three layer 1 blocks. Temporal masking technology is used to get masking threshold in psychoacoustic model I. The theoretical minimum encoding and decoding delay for Layer II is about 35 ms [1]. The compression ratio for layer 2 is 1:6 to 1:8 and the data rates for layer 2 is from 56 to 193 kbps [6].

MPEG-1 layer 3 has the most complicated encoder and decoder compared to layer 1 and layer 2. Unlike layer 1 and layer 2, it introduces increased frequency resolution based on a hybrid filterbank, which is a serial combination of subband filterbank and

Modified Discrete Cosine Transform (MDCT). A hybrid filterbank divides original audio into 32 equal frequency subbands. Every band is split into 18 frequency lines by use of an MDCT. Dynamic window switching is used to control time artifacts (pre-echoes). Shorter blocks are used can be selected for better time resolution and longer blocks are usually used for better frequency resolution. Layer 3 adds a different quantizer (nonuniform), adaptive segmentation and entropy coding of the quantized values. Temporal masking technology is used to get masking threshold in psychoacoustic model, but psychoacoustic model II is used in layer 3. Unlike psychoacoustic model I, model II never actually separates tonal and non-tonal components. Instead, it computes a tonality index as a function of frequency. This index gives a measure of whether the component is more tone-like or noise-like, which keeps more useful audio signal and helps enhance the quality of the audio. The theoretical minimum encoding and decoding delay for Layer 3 is about 19 ms [1]. The compression ratio for layer 3 is 1:10 to 1:12 and the data rates for layer 3 is between 128 to 112 kbps [6]. Table 2.1 [7] shows the features of the three layers.

Table 2.1: Features of MPEG-1 layers 1, 2, and 3

	Layer I	**Layer II**	**Layer III**
Filterbank	Polyphase	Polyphase	Hybrid
Output bit-rate	32-448 kbps	32-384 kbps	32-320 kbps
Efficient bit-rate	160-224 kbps	96-128 kbps	64-96 kbps
Sampling frequency	32, 44.1, 48 kHz	32, 44.1, 48 kHz	32, 44.1, 48 kHz
Intensity stereo	Yes	Yes	Yes
Quantization	Uniform	Uniform	Non-Uniform
Window	Fixed	Fixed	Dynamic
Entropy coding	No	No	Yes
Frame size	384 samples	1152 samples	1152 samples
Bit-allocation representation	Explicit	Indexing	Indexing
Suggested psychoacoustic model	Model I	Model I	Model II

For a given sound quality level, MP3 requires the lowest bit rate. Or, for a given bit rate, it achieves the highest sound quality. Table 2.3 [6] shows some typical performance

Table 2.2: Compression ratios and data rates for MPEG-1 layers 1, 2 and 3

Layers	Ratio	Bit Rate
Layer 1	1:4	384 kbps
Layer 2	1:6 ... 1:8	56 ... 192 kbps
Layer 3	1:10 ... 1:12	128 ... 112 kbps

of MP3.

Table 2.3: Some typical performance data of MPEG-1 layer 3

Sound quality	Bandwidth	Mode	Bit rate	Reduction ratio
Telephone sound	2.5 kHz	mono	8 kbps	96:1
Better than short wave	4.5 kHz	mono	16 kbps	48:1
Better than AM radio	7.5 kHz	mono	32 kbps	24:1
Similar to FM radio	11 kHz	stereo	56 to 64 kbps	26:1 to 24:1
Near-CD	15 kHz	stereo	96 kbps	16:1
CD	15 kHz	stereo	112 to 128 kbps	14:1 to 12:1

As a result, MP3 allows us to compress better while keeping higher sound quality. Therefore, MP3 compressed audio is the most used audio format in industry. In this thesis, the research is on new digital watermarking algorithms for MP3 compressed audio signals.

2.2 MP3 in Detail

The invention of the MPEG-1 Audio Layer-3 (MP3) audio compression algorithm is probably one of the most remarkable developments in the area of digital media processing because of its high compression ratio and quality. MP3 creates a highly compressed signal with high perceptual quality by eliminating or reducing unnecessary frequencies.

A complete MP3 system has an MP3 encoder and an MP3 decoder. The operations of the encoder and decoder is governed by a set of undetermined control signals. These are combination of both external hardware controls and software functions. The overall

MP3 system architecture is illustrated in Figure 2.1 [8].

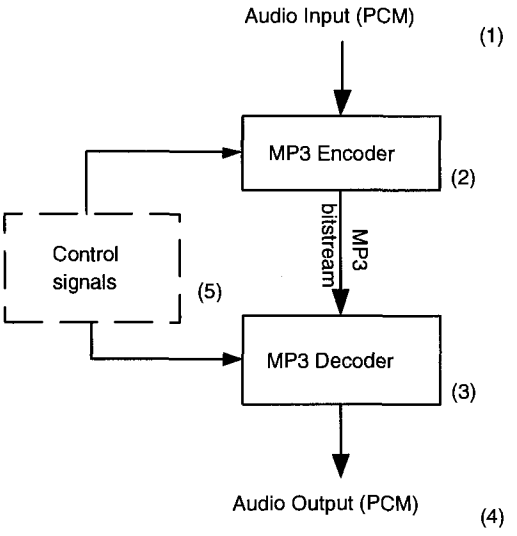

Figure 2.1: An overview of MP3 system architecture.

In an MP3 system, a regular PCM audio is input to the MP3 encoder. This analog signal will be converted to an MP3 digital bitstream according to the standard MPEG-1 layer 3 encoding algorithm. An MP3 file will be generated after MP3 encoder. The decoder reconstructs the analog audio signal by decoding the MP3 bitstream, and the analog audio signal is finally turned into audible sounds. The control signals mean external software functions and parameter controls during MP3 encoding and decoding.

In the next sections, MP3 encoder and decoder will be introduced in depth.

2.2.1 MP3 Encoder

Figure 2.2 shows the block diagram of a typical MP3 encoder. The following describes each of the components.

FFT Analysis A 1024 point Fast Fourier Transform (FFT) supplies the psychoacoustic model with the frequency components of the input signal.

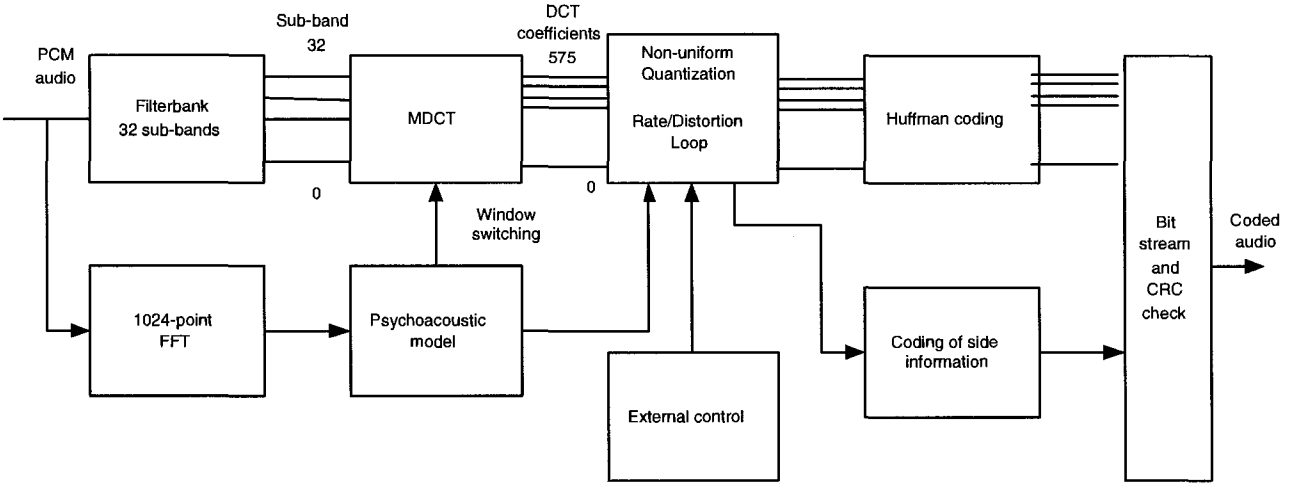

Figure 2.2: Block diagram of an MP3 encoder.

Filterbank The purpose of the filterbank is to transform audio signals to frequency domain signals in 32 subbands. Every subband is split into 18 frequency lines. Therefore, the 32 subbands are then mapped into 576 (32×18) frequency lines after Modified Discrete Cosine Transform (MDCT). Empirical evidence has shown that the human ear has a limited resolution (that can be expressed in terms of critical bandwidths) less than 20 kHz and more than 20 Hz, which is the range of Human Auditory System (HAS). Within a critical bandwidth the human ear blurs frequencies. Thus the filterbank creates equal-width frequency subbands that correlate to the critical bandwidths.

Compared with MPEG-1 layer 1 and layer 2 that use polyphase filterbank, MPEG-1 Layer 3 (MP3) belongs to the class of hybrid filterbanks. For hybrid filterbank, an additional Modified Discrete Cosine Transform (MDCT) is used in Layer 3. The signal in each bank can be encoded differently depending on its energy and contribution to the perceptual quality of the output audio. Polyphase filterbank is lossy but without quantization the MDCT transformation is lossless. The

Hybrid FilterBank adapts to the signal characteristics with block switching. Layer 3 specifies two different MDCT block lengths, one is long block with a higher frequency resolution and the other one is short block with higher time resolution. Layer 3 uses mixed blocks where the two first subbands are long block structure, and the upper bands are short block structure. Mixed block is generated in layer 3 only. Block switching is triggered by psychoacoustic model.

Psychoacoustic Model The human ear can nominally hear sounds in the range of 20 Hz to 20 kHz. Psychoacoustic model analysis is the technology on HAS to make critical banks, which is not the same as 32-equal subbands after filterbank. When in low frequency, a subband might include several critical bands, however, when in high frequencies, several subbands are included into one critical band. Unlike 32 subbands, critical bands are non-linear algorithms. Psychoacoustic model uses the weakness of the human ear to eliminate the inaudible parts of signals to make compression by using masking technology on critical bands. There are two masking technologies usually used. One is frequency masking and the other one is temporal masking. Temporal masking technique is used in MP3. Temporal masking is the characteristics of the auditory system where sounds are hidden due to maskers which have just disappeared or about to appear. The effect of masking after a strong sound is called post-masking. This is different from pre-masking, where a sound actually is masked by something which appears after it. Therefore, in MP3 it is important to calculate the audio signal strength (energy) on scalefactor bands (critical bands) for watermarking.

Unlike psychoacoustic model I, psychoacoustic model II never actually separates tonal and non-tonal components. Instead, it computes a tonality index as a

function of frequency. This index gives a measure of whether the component is more tone-like or noise-like. Model II uses this index to interpolate between pure tone-masking-noise and noise-masking-tone values. The tonality index is based on a measure of predictability. Tonal components are more predictable and thus will have higher tonality indices. Because this process relies on more data, it is more likely to better discriminate between tonal and non-tonal components than the model I method [1]. If subbands are wider than critical bands, minimal masking threshold will be used in subband. Otherwise, average masking threshold in subband will be used [9]. Compared to model I, model II has the same accuracy for the higher subbands as for low frequency because it does not concentrate the non-tonal components.

In this research, MP3 audio energy in each critical band (scalefactor band) is calculated by using psychoacoustic model II and then mapped to MDCT values in each frame according to the tables for converting MDCT index to scalefactor bands. The energy value is calculated to be one of the watermarking parameters in the proposed watermarking algorithms.

MDCT and Window Switching MDCT (Modified Discrete Cosines Transform) is lossless and is performed on the consecutive blocks of a larger dataset, where subsequent blocks are overlapped so that the last half of one block coincides with the first half of the next block. This overlapping, in addition to the energy-compaction qualities of the DCT, makes the MDCT especially attractive for signal compression applications, since it helps to avoid artifacts stemming from the block boundaries [10]. The MDCT is performed on blocks that are windowed and overlapped 50 percentage off.

The MDCT is normally performed on 18 samples at a time (long blocks) to achieve good frequency resolution. It can also be performed on 6 samples at a time (short blocks) to achieve better time resolution and to minimize pre-echoes. There are special window types for the transition between long and short blocks.

Compared to MPEG-1 layers 1 and 2, layer 3 has mixed blocks to avoid the per-echo problem. Pre-echo is a psychoacoustic phenomenon where an unusually noticeable artifact is heard in a sound recording from the energy of time domain transients smeared backwards in time after processing in the frequency domain because forward temporal masking is so much stronger than backwards temporal masking, the sounds like an echo which comes before the actual sound. In mixed blocks, the first two subbands are using long block structure and the rest of subbands use short block structure. Long block is good at lower bands and short block is good at upper bands. Mixed blocks are useful to reduce pre-echo in case of transients, while keeping a good frequency resolution in the lower part of the spectrum. In mixed blocks structure, the two first subbands control the time domain signal energy and the upper bands contain the information after processing in the frequency domain [1].

Quantization and Coding After MDCT, a non-uniform quantization maps amplitude values into finite number of bits. Non-uniform is to change sample size according to amplitude values [11]. In this way, larger values are automatically coded with less accuracy and some noise shaping is already built into the quantization process. Quantization is the second time compression in MP3 file generation. There are two-nested loop during the quantization step. One is inner iteration loop (to control quantization step size) and the other one is outer iteration loop

(to control the noise shaping factors for each scale factor band). Actual quantization of MP3 is performed in the inner iteration loop doing the rate control. This inner iteration loop is first done in the outer iteration loop, which controls the distortion.

Inner Interaction Loop (Rate Loop) Inner interaction also named as rate loop to make smaller quantized values according to Huffman code tables. If the number of bits resulting from the coding operation exceeds the number of bits available to code a given block of data, this can be corrected by adjusting the global gain to result in a larger quantization step size, leading to smaller quantized values. This operation is repeated with different quantization step sizes until the resulting bit demand for Huffman coding is small enough. The loop is called rate loop because it modifies the overall coder rate until it is small enough.

Outer Interaction Loop (Noise Control/Distortion Loop) Outer interaction loop (also called as noise control loop) shapes the quantization noise according to the masking threshold by psychoacoustic model II. Scale factors are applied to each scale factor band. If the quantization noise in a given band is found to exceed the masking threshold (allowed noise) as supplied by the perceptual model, the scale factor for this band is adjusted to reduce the quantization noise. Since achieving a smaller quantization noise requires a larger number of quantization steps and thus a higher bit rate, the rate adjustment loop has to be repeated every time new scale factors are used. In other words, the rate loop is nested within the noise control loop. The outer (noise control) loop is executed until the actual noise (computed from the difference of the original spectral values minus the quantized spectral values) is below the masking threshold for

every scalefactor band (critical band) [1].

Figure 2.3 shows the relationships between inner loop and outer loop [12].

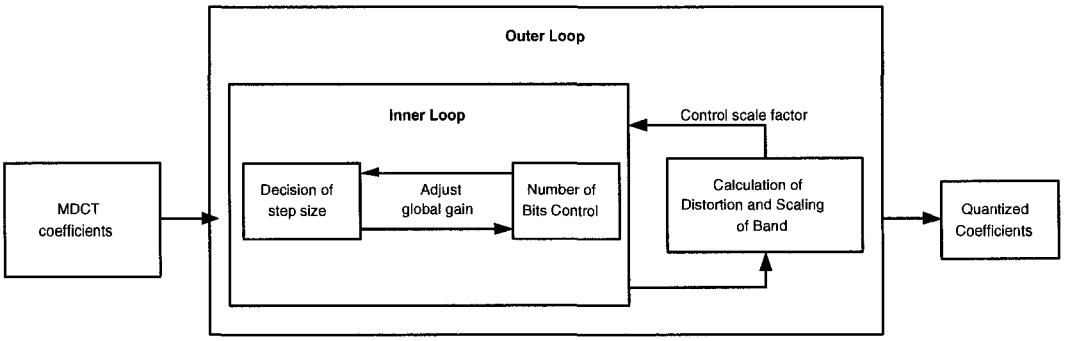

Figure 2.3: The relationships between inner loop and outer loop.

In short, the outer loop monitors the distortion of each scalefactor band to check if it is greater than the allowed distortion provided by the psychoacoustic model II. If so, more bit are assign to this scalefactor band by varying its scalefacotor. The inner loop determines the global gain of the granule based on the available bits. If the quantized spectral values require more bits than available ones, the global gain is adjusted accordingly to reduce the required bits [13]. Therefore, once the global gain is changed in inner loop, the distortion of each scalefactor band is also required to change in outer loop. As the result, the outer loop must be executed again for distortion control when the global gain in inner loop is changed. Similarly, if varying the scalefactor of scalefactor band affects the overall encoding bits for the granule, the inner loop needs to be executed again. Thus, MP3 quantization is also named as two-nested loop quantization.

Side Information Side information records the information in the bitstream necessary, such as Huffman table selection, window switching and quantization control for controlling the decoder.

Huffman Coding and Bitstream Generation The quantized values are stored by Huffman coding. As a specific method for entropy coding, Huffman coding is lossless and keeps data in the bitstream along with the scale factors and side information. Quantized MDCT coefficients are arranged in the order from lowest to highest frequency. The whole range is divided into three sections, each coded with a different set of Huffman tables. Bit generation and CRC check make the signal frames finally to be MP3 bitstream.

External Control (Bit allocation) Through an iterative algorithm, the bit allocation uses information from the psychoacoustic model to determine the number of code bits to be allocated to each subband. This process can be described using the following formula:

MNRdB = SNRdB - SMRdB

where, MNRdB is the mask-to-noise ratio; SNRdB is the signal-to-noise ratio, given by the MPEG-1 standard; and SMRdB is the signal-to-mask ratio, derived from the psychoacoustic model.

Then the subbands are placed in order of lowest to highest mask-to-noise ratio, and the lowest subband is allocated the smallest number of code bits and this process continues until no more code bits can be allocated. Two nested iteration loops called the rate loop and the distortion loop serve to quantize and code in MP3 encoders.

2.2.2 MP3 Frame

MP3 encoded bitstream is divided into frames. The frame is a central concept when decoding MP3 bitstream. Therefore, watermarking on MP3 frames is the major con-

tribution of this thesis. An MP3 frame consists of five parts including header, CRC, side information, main data and ancillary data. Figure 2.4 shows MP3 frame structure. The size of each frame is usually fixed.

A frame usually consists of 1 or 2 granules. Each granule is further divided into 32 subband blocks of 18 frequency lines. Therefore, in stereo audio, each frame includes two granules which are left and right granules and each granule is a 32 subband block of 18 frequency lines. For mono audio, a frame only includes one granule. Therefore, in mono audio a frame is the same as a granule. In this research, the proposed MP3 watermarking algorithm is a generic method for MP3 frames. However, in order to easily present, the proposed watermarking algorithm is implemented on one granule in a frame. Therefore, for stereo audio, the new algorithm is implemented on its right granules in a frame; for mono audio, the proposed algorithm is implemented on its entire frame. The experimental results of the proposed watermarking algorithm are on the mono audio only.

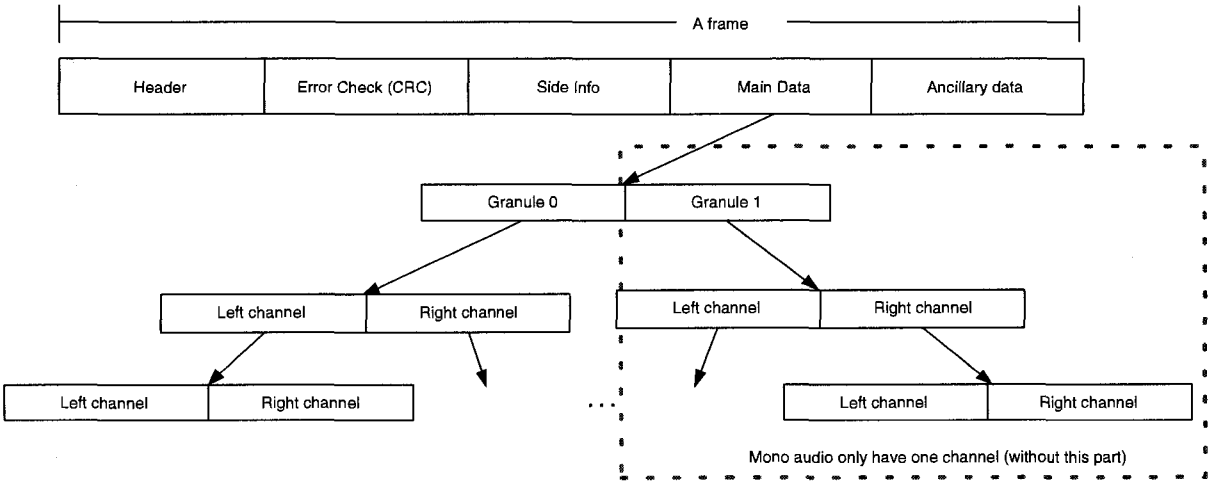

Figure 2.4: MP3 frame structure.

The size of an MP3 frame depends on the audio bit rate and sampling rate. The

total size in bytes for any given frame can be calculated with the following formula:

$FrameSize = 144 \times BitRate/(SampleRate + Padding)$.

For example, if encoding a file at 128 kbps, the original sample rate is 44.1 kHz, and no padding bit has been set, the total size of each frame will be 418 bytes $(144 \times 128000/(44100 + 0))$.

The following explains each of the MP3 frame components.

Header For each frame, the size of header is fixed. It is a 32-bit 0/1 stream, which includes the sync word and ancillary data for decoder. The breakdown information of an MP3 frame header is shown in Table 2.4 [14].

Table 2.4: MP3 frame header

Field	Length (in bits)
Frame sync	11
MPEG audio version (MPEG-1, 2, etc.)	2
MPEG layer (Layer I, II, III, etc.)	2
Protection (if on, then checksum follows header)	1
Bitrate index (lookup table used to specify bitrate for this MPEG version and layer)	4
Sampling rate frequency (44.1 kHz, etc., determined by lookup table)	2
Padding bit (on or off, compensates for unfilled frames)	1
Private bit (on or off, allows for application-specific triggers)	1
Channel mode (stereo, joint stereo, dual channel, single channel)	2
Mode extension (used only with joint stereo, to conjoin channel data)	2
Copyright (on or off)	1
Original (off if copy of original, on if original)	1
Emphasis (respects emphasis bit in the original recording; now largely obsolete)	2

Error Check In MP3, frame uses 16 bit CRC to do error check as an optional feature by users. This error check is to check the most sensitive data for transmission errors. Sensitive data is defined by the standard to be both header and side information. If these values are incorrect they will corrupt the whole frame whereas an error in the main data only distorts a part of the frame. A corrupted frame can either be muted or replaced by the pervious frame [15].

Side Information The side information part of the frame consists of information needed to decode the main data. The size depends on the encoded channel mode. If it is a single channel bitstream the size will be 17 bytes; if not, 32 bytes are allocated. The different parts of the side information are presented in Figure 2.5 and described in detail below. In this thesis, mono mode is used, which has one channel only.

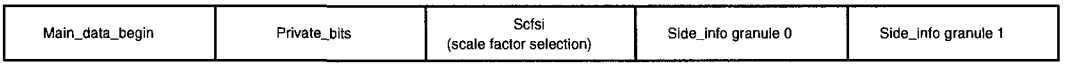

Figure 2.5: MP3 frame side information.

Main Data MP3 frame main data are made up by granules. For mono audio, which has one channel, the frame main data include only one granule. For stereo audio, which has two channels, a frame main data include two granules. Figure 2.6 shows the format of MP3 frame, granules, subband blocks and frequency lines.

Each frame is divided into one (mono) or two (stereo) granules. Each granule contains 32 subbands, and each subband contains 18 frequency lines (MDCT values). Figure 2.7 shows a granule in detail.

Ancillary Data In MP3 frame, ancillary data are optional and user definable. The number of ancillary bits equals the available number of bits in an audio frame minus the number of bits actually used for header, error check and audio data. In Layer III, the number of ancillary bits corresponds to the distance between the end of the Huffman code bits and the location in the bitstream where the next main data beginning pointer points to.

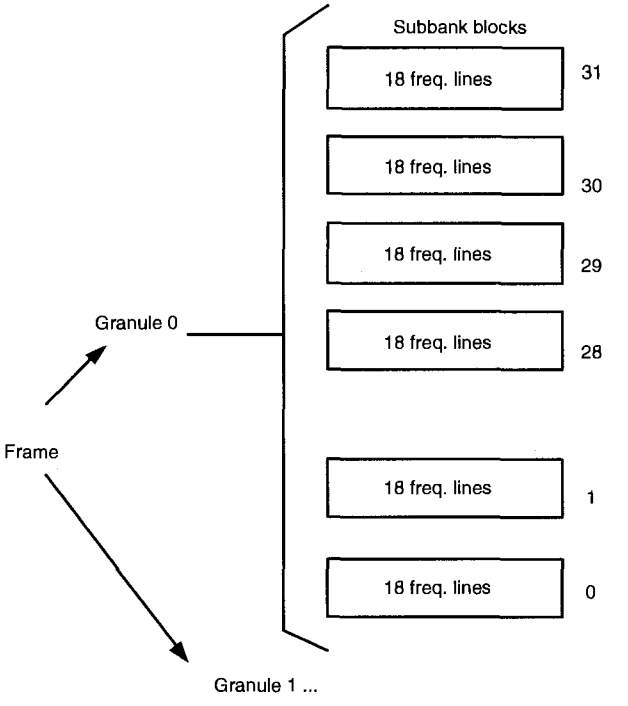

Figure 2.6: Format of MP3 frame, granules, subband blocks and frequency lines.

2.2.3 MP3 Decoder

In MP3 decoder, most of the steps are the inverse of the encoder, but there is no psychoacoustic modeling in decoder. Figure 2.8 [1] shows the block diagram of a typical MP3 decoder. And the components in an MP3 decoder are explained in the following.

Huffman Decoding The Huffman decoder translates the variable length codes in the MP3 bitstream to spectral lines. The decoder uses 32 fixed tables from the

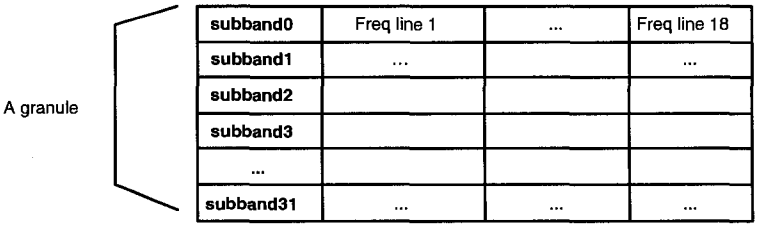

Figure 2.7: A granule.

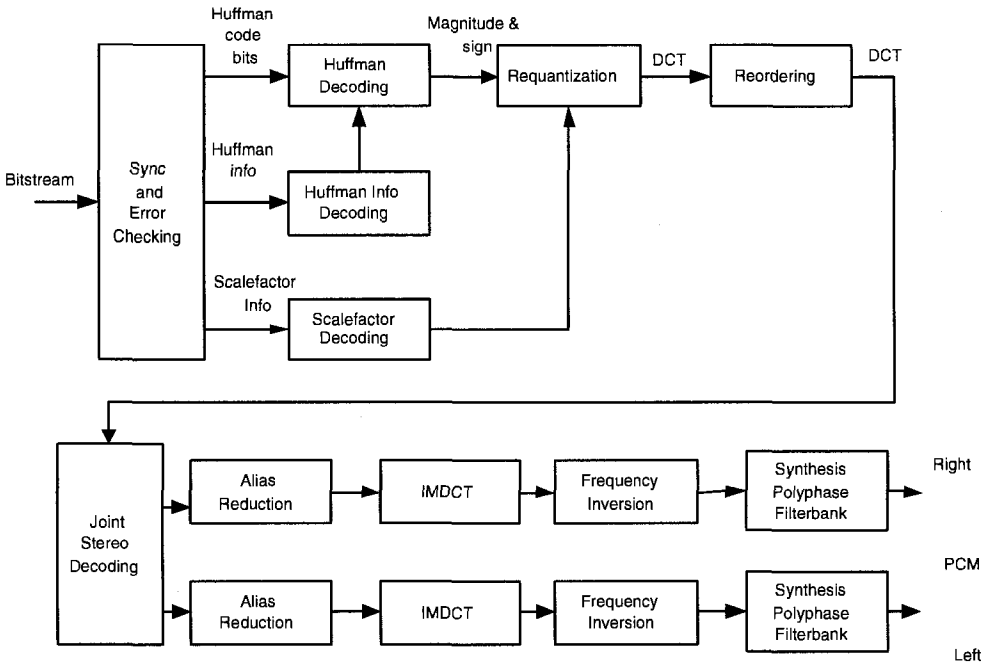

Figure 2.8: MP3 decoder structure.

standard that contain information about how the codes are designed.

The bitstream side information is used to select the tables for the different parts of the frequency spectrum.

There is a special mechanism in the decoder for decoding large values. Certain decoded values imply that a table dependent number of linear bits must be read and added to the value. Sign bits also have special handling.

The longest variable length codeword for any table is 19 bits. There are only 16 different tables actually defined by the standard.

Re-quantization and Re-scaling The sample re-quantization block uses the scale factors to convert the Huffman decoded values back to their spectral values.

The requantized samples must be reordered for the scale factor bands that use

short windows. In this example there are a total of 18 samples (i.e. a1 to c6) in a band that contains 3 windows (a-c) of 6 samples each (1-6). Figure 2.9 [1] illustrates the reordering of samples.

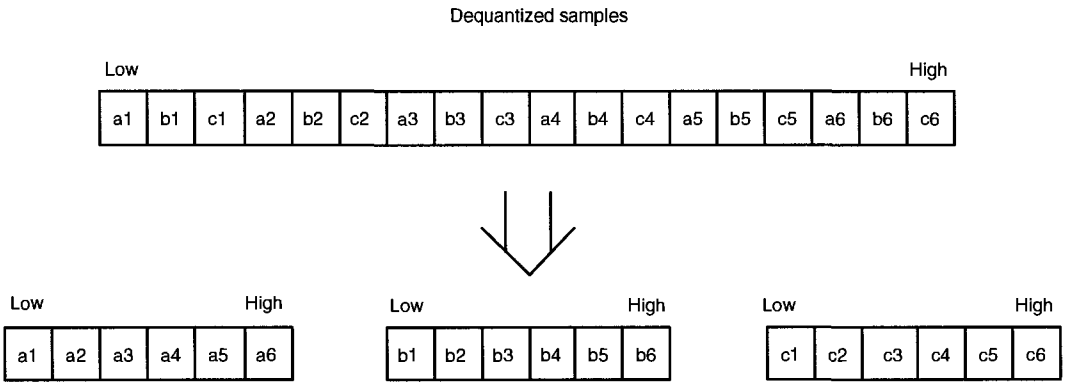

Figure 2.9: Reordering of samples.

The short windows are reordered in the encoder to make the Huffman coding more efficient, since the samples close in frequency (low or high) are more likely to have similar values.

Stereo Decoding The compressed bitstream can support one or two audio channels in one of four possible modes:

1. A monophonic mode for a single audio channel.
2. A dual-monophonic mode for two independent audio channels (functionally identical to the stereo mode).
3. A stereo mode for stereo channels that share bits but do not use joint-stereo coding.
4. A joint-stereo mode that takes advantage of either the correlations between the stereo channels (MS (Mid-Side) stereo) or the irrelevancy of the phase difference between channels (intensity stereo), or both.

The stereo processing is controlled by the mode and mode extension fields in the frame header.

Alias Reduction The alias reduction is required to negate the aliasing effects of the filterbank in the encoder. It is not applied to granules that use short blocks.

The alias reduction consists of eight butterfly calculations for each subband as illustrated by the Figure 2.10 [1].

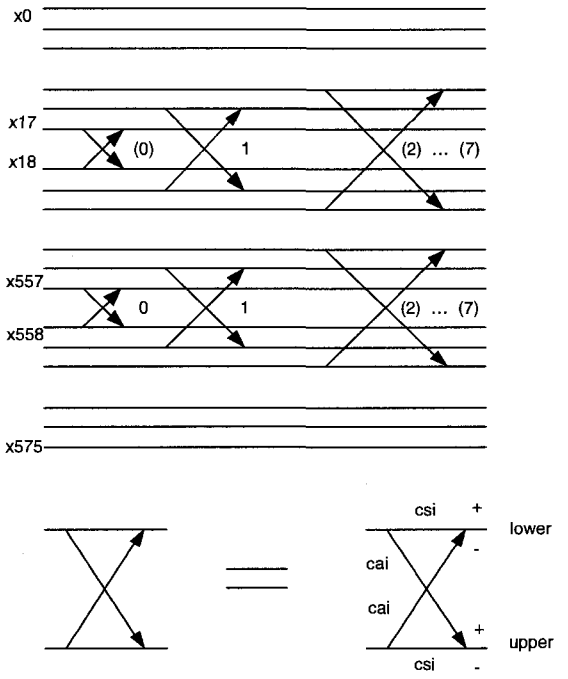

Figure 2.10: Alias reduction butterflies. [1]

Our proposed adaptive watermarking algorithm extracts watermark after the alias reduction step before the IMDCT step.

IMDCT The IMDCT (Inverse Modified Discrete Cosine Transform) transforms the frequency lines to polyphase filter subband samples. The output from the IMDCT operation is 18 time-domain samples for each of the 32 subband blocks.

Frequency Inversion In order to compensate for frequency inversions in the synthesis polyphase filterbank every odd time sample of every odd subband is multiplied with -1. The subbands are numbered [0..31] and the time samples in each subband [0..17].

Synthesis Polyphase Filterbank The synthesis polyphase filterbank transforms the 32 subband blocks of 18 time domain samples in each granule to 18 blocks of 32 PCM samples.

The filterbank operates on 32 samples at a time, one from each subband block, as illustrated by Figure 2.11 [1].

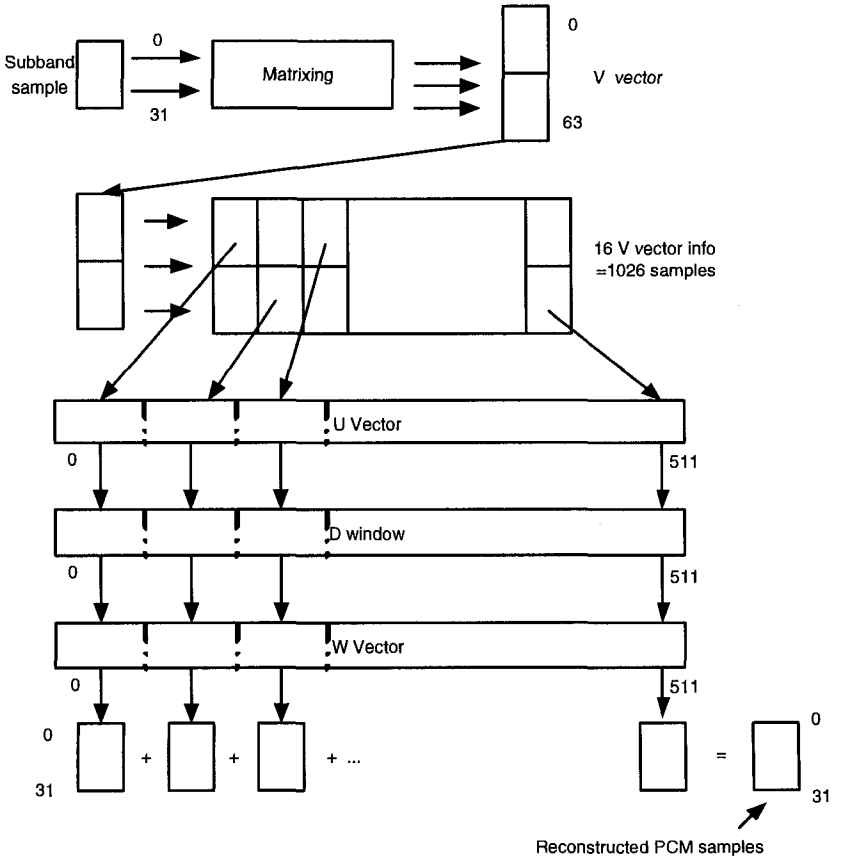

Figure 2.11: Synthesis polyphase filterbank. [1]

In the synthesis operation, the 32 subband values are transformed to a 64-value V vector using a variant of the IMDCT. The V vector is pushed into a FIFO that stores the last 16 V vectors.

A U vector is created from the alternate 32 component blocks in the FIFO as illustrated and a window function D is applied to U to produce the W vector. The reconstructed samples are obtained from the W vector by decomposing it into 16 vectors each 32 values in size and summing these vectors to generate the reconstructed PCM samples.

In this research, we embed watermark into MP3 frames during MP3 compression. In the proposed adaptive watermarking algorithms, the statistic analysis (Gaussian distribution) is introduced to calculate the MP3 frame energy to adapt the watermark coefficients dynamically to the original audio strengths. Blind watermark detection is implemented according to the watermark embedding method. Chapter 4 will present the proposed watermarking algorithms in detail.

Chapter 3

Overview of Digital Audio Watermarking

3.1 Digital Watermarking

First digital watermarking related publications date back to 1979 by Szepanski, who described a machine-detectable pattern that could be placed on documents for anti-counterfeiting purposes. However, it did not gain an international interest until 1990s [3]. Nowadays, digital watermarking of multimedia content has become a very active research area both in academy and in industry.

The basic idea of digital watermarking is to describe methods and technologies that allow the hiding of information named watermark, imperceptibly into original multimedia data, such as image, audio and video. Ideally, there should be no perceptible difference between the watermarked and original signal, and the watermark should be robust enough for difficulty to remove or alter without damaging the host signal.

In general a watermarking system includes an embedder and detector, as illustrated

in Figure 3.1.

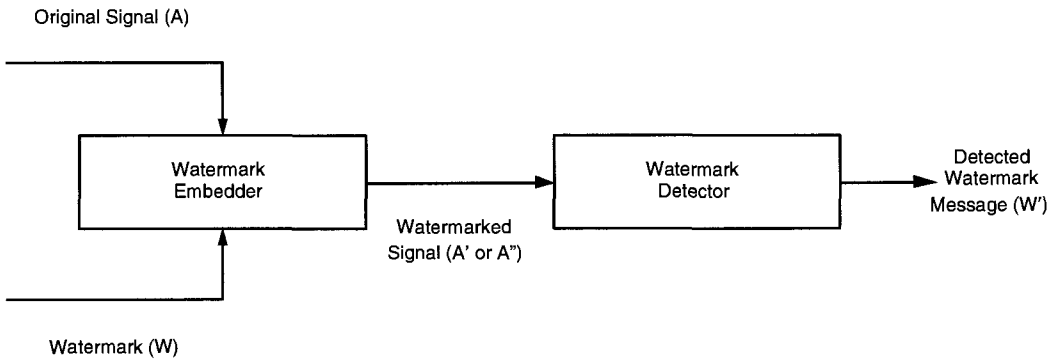

Figure 3.1: A generic watermarking system.

An embedder usually embeds watermark (W) into the original signal (A). The watermark, a stream of data, can be looked as the security key for copyright protection, labeling, monitoring, tamper proofing and conditional access. The watermark W is embedded into original signal A to form the watermarked signal A'. The embedding algorithm is defined as Equ. (3.1) [16].

$$A' = E(A, W) \tag{3.1}$$

Watermarked signal A' might change to be watermarked signal A" through various manipulation such as signal distortion, cropping, malicious attacks, signal compression, filtering or random noise.

Watermark extraction obtains the possibly distorted watermark W'. W' might not be the same as the original watermark W because of distortions. The watermark extraction process can be illustrated by Equ. (3.2) [16].

$$W' = D(A", A) \tag{3.2}$$

where A is optional to watermark extraction.

We can use the extracted watermark W' to compare with the original watermark W to achieve our purpose of copyright protection or content authentication.

In general, a watermark must be robust to deliberate attacks and inadvertent distortion during all sorts of manipulations. Ideally it is also transparent to the user of the material so that it does not perceptibly distort original content.

3.2 Properties of Digital Watermarking

A digital watermarking system has several of properties based on the different applications but as a successful watermarking system, it should satisfy a set of the following basic characteristics.

3.2.1 Undetectability

The watermarks are designed to be resistant to attempts at forgery, alteration, erasure and decoding by unauthorized parties. A watermark has to be statistically undetectable and accessible by unauthorized persons in order to prevent its removal [17]. Only those who know the specified key can extract watermarks hidden in files.

3.2.2 Imperceptibility

The watermark should not affect the quality of the original signal. Therefore, imperceptibility is a basic requirement of digital watermarking to prevent its visual or aural identification and ensures the preservation of the original signal quality.

This is usually achieved by implicitly or explicitly taking into account the characteristics of the human visual or auditory system [17]. In image watermarking, watermark

data may be hidden in images by exploiting the properties of the human visual system (HVS). In audio watermarking system, we usually use perceptual masking model to control and generate the watermark based on human auditory system (HAS) to hide the copyright information into the original audio signal [18].

Perceptual quality are usually used to evaluate the imperceptibility of a watermarking method. Perceptual quality refers to the imperceptibility of embedded watermark data within the host signal. The signal-to-noise ratio (SNR) and peak signal-to-noise ratio (PSNR) of the watermarked signal versus the host signal can be used as a quality measure. In this research, we use SNR and PSNR to evaluate audio quality.

Signal-to-noise ratio (SNR) is defined by Equ. (3.3) [3]:

$$SNR(f,g) = 10\log_{10}\frac{\sum_{\forall i} f(i)^2}{\sum_{\forall i}(f(i)-g(i))^2} \tag{3.3}$$

where f is regarded as the original audio signal and g is respectively the watermarked signal and N is the length of the audio. The distance measures how noisy f is in relation to g.

The peak signal-to-noise ratio (PSNR) of the watermarked signal versus the original signal is defined by Equ. (3.4) [19].

$$PSNR(f,g) = 10\log_{10}\frac{(\max_{\forall i}(f(i)))^2}{\frac{1}{N}\sum_{\forall i}(f(i)-g(i))^2} \tag{3.4}$$

where f and g are respectively the original audio and the watermarked audio, and N is the length of the audio.

In addition to SNR and PSNR, we also use Perceptual Evaluation of Speech Quality (PESQ) to evaluate audio quality. PESQ is a mechanism for automated assessment of speech quality. It is standardized as ITU-T recommendation P.862 [20]. PESQ directly

uses MOS (Mean Opinion Score) to express the speech quality. The PESQ MOS as defined by the ITU recommendation P.862 ranges from 1.0 (worst) up to 4.5 (best) [20].

3.2.3 Robustness

A watermark has to be robust to signal manipulation and its removal should be impossible without perceptible signal alteration [17]. The robustness should extend to common signal processing operations, such as linear filtering, sample re-quantization, various conversion and lossy compression.

In this research, robustness is mainly measured by the bit error rate (BER) of extracted watermark data as a function of the amount of distortion introduced by a given operation. Bit rate refers to the amount of watermark data that may be reliably embedded within a host signal per unit of time or space, such as bits per second or bits per pixel. Reliability is measured as the bit error rate (BER) of extracted watermark data. The bit error rate is computed by:

$$BER = \frac{\sum_{j=1}^{N} w(j) \bigoplus w_i^*(j)}{N} \tag{3.5}$$

where w is the original watermark, w^* is the extracted watermark, N is the length of the watermark, and $\bigoplus$ is exclusive-OR operator.

3.2.4 Capacity

Capacity is the number of bits that can be reliably embedded into the original signal. The capacity of a watermarking algorithm must be sufficient to enable the envisioned application [21].

3.2.5 Computational Complexity

Digital watermark is also restricted by computational complexity of embedding and decoding. The actual CPU time (in seconds) consumed by an algorithm is one of the indicators to show the speed of the method. In some real-time application, the computational complexity requirement is raised. For instance, broadcast monitoring system asks for real-time operation, therefore, the embedding and decoding systems have to be able to deal with the related information in real time.

3.3 Applications of Digital Watermarking

Digital watermarks can be used for various applications. Different applications use different watermarking properties. We introduce the following most popular and widely used applications.

3.3.1 Copyright Protection

Copyright protection is the original goal of a digital watermarking system. Watermarks for copyright protection generally fall into two categories. One is proof of ownership and the other one is the enforcement of usage policies [22].

Proof-of-ownership watermarks are secret watermarks, embedded into original media using a secret key without which a detector cannot determine its presence [22]. These marks are intended to be evidence of rightful ownership perhaps in a court of law.

The enforcement of usage policies is an application much desired by the recoding and motion picture industries. In this well-known application, watermarks provide instructions or copyright information that can be read by set-top boxes, MP3 players,

CD-burners and other consumer products. This information directs devices to honor permissions associated with the music, perhaps refuse to record or even play music in violation of a usage policy.

3.3.2 Fingerprinting

Fingerprinting is a form of watermarking whose goal is the tracing of an audio clip's usage history. A fingerprinting system would embed a unique ID in each copy of a piece of digital music, so that a pirated song could be traced to the individual who purchased it. This could be used to more effectively prosecute copyright infringement and perhaps control the illegal music downloading [22].

3.3.3 Content Authentication

Signature information is embedded into the original signal to verify and prevent unauthorized alteration of contents. With the Internet industry becoming more popular, many companies provide a web spider service that routinely searches the web and tests for digital files that have been watermarked. The purpose of this service is to detect violations of copyright for registered content owners. Once the copyright violation is detected, content owners will be notified and the infringer will be likely taken to court [23].

3.3.4 Access Control

Watermarks can be used for data annotation and access control, which prevents unauthorized copy, playback of multimedia. The control of content usage is enabled through the user's player application. When the digital content is accessed in a compliant user

device, the user's player application counts annotation watermarks, checks the usage restrictions and updates watermarks as required. If the amount of annotation watermarks does not comply with the counting usage, the user's device will not perform the request. The major advantage of using annotation watermarks is that it binds usage rights with digital content no matter where the content travels [23].

3.3.5 SDMI

The Secure Digital Music Initiative (SDMI) is a forum that brings together the worldwide recording, consumer electronics and information technology industries to develop open technology specifications for protected digital music distribution [24]. The specifications released by SDMI will ultimately provide consumers with convenient access to music both online and in new emerging digital distribution systems, enable copyright protection for artists' work, and promote the development of new music-related businesses and technologies.

In industry, SDMI has selected an audio watermarking technology that will be integrated into the next generation of portable digital Internet music devices, including MP3 players.

3.4 Attacks to Digital Watermarking

Ideally, an effective, robust watermarking scheme provides a mark that can only be removed when the original content is destroyed as well. The degree of robustness and distortion necessary to alter the value of the original content can vary for different applications. In robust watermarking applications, attacker makes the detector unable to detect the watermark while keeping the perceptual quality. In fragile watermarking

applications, attacker makes the watermark still valid after alteration of work. Usually, Human Auditory System (HAS) is more sensitive than Human Visual System (HVS) in applications.

There are several attacker assumptions: attacker knows nothing, attacker has more than one watermarked work, attacker knows the algorithm, and attacker has access to the detector as an oracle [25]. In general, there are several common types of attacks from attackers, which are unauthorized embedding, unauthorized detection, unauthorized removal, and system attacks [3].

3.4.1 Unauthorized Embedding

The most common unauthorized embedding attacks is against composing and embedding an original message. For instance, an attacker forges a valid watermarked work for new host data instead of the original watermark. Or, an attacker copies blocks of valid work without understanding the content [25]. To use cryptographic tools or digital signature to prevent forging is an approach to stopping unauthorized embedding attacks.

3.4.2 Unauthorized Detection

Attackers decode the watermark content with the existence of a watermark. In some applications, watermark detection should be restricted. For example, a hospital might embed the patients ID into their X-ray. The reason for privacy is to prohibit unauthorized individuals from reading the marks and identifying the patients [3]. In this case, unauthorized detection attacks should be avoided. To encrypt watermark before embedding is a way to prevent unauthorized detection attacks from attackers.

3.4.3 Unauthorized Removal

In all watermarking applications that require security against unauthorized removal, there are two kinds of unauthorized removal attacks often used. One is elimination attack, which removes the watermark, no one can detect it. The other one is called masking attack, which means the watermark remains present, but only a smart detector can detect it. We can define elimination attacks as those that move watermarked works into regions of media space where unwatermarked works are likely, and masking attacks as those that move watermarked works into regions in which unwatermarked works are not likely [3].

3.4.4 System Attacks

Attacks that exploit the weakness of how the watermark is used can be broadly referred to as system attacks [3]. As a simple example of a system attack, consider a copy-control application in which every recording device includes a computer chip for detecting the watermark. An adversary might open the recoder and just remove the chip, thereby causing the recorder to allow illegal copying. In such an attack, the security of the watermark itself is defeated [3].

3.5 Literature Review on Audio Watermarking

Audio watermarking also requires imperceptibility (inaudibility), and robustness to distortions and attacks. In this section, the most common used audio watermarking techniques are reviewed.

Low-bit coding is the simplest way to embed watermark into audio in spatial domain. The basic idea in this technique is to replace the least significant bit (LSB)

of each sampling point by a coded binary string. For example, in 16-bits per sample representation, the least four bits can be used for hiding [16]. Thus, a large amount of data can be embedded into an audio signal with this approach. Low-bit coding is easy to embed a large amount of data into audio signal. But it has poor immunity to manipulations. The embedded watermark can be destroyed by many signal processing attacks easily.

Gruhl et al. [26] proposed echo-based watermarking method, which was to add a repeated version of a component of the audio signal with small enough offset, initial amplitude, and decay rate to make it imperceptible. The echo hiding encoder embeds data by introducing an echo. The echo was an imperceptible information period into an audio data stream. There are four major parameters for an echo: initial amplitude, decay rate, one offset, and zero offset. The original signal and the echo with one offset or zero offset are combined during watermark embedding. When watermark detection, watermark is extracted based on the original signal and offset correlation. The decision of the time delay can be made by examining the position of a spike that appeared in the autocorrelation diagram. Echo hiding can effectively embed imperceptible information into an audio signal [27]. However, since the Echo hiding method relies on decay rate and offsets of the original signal to embed watermark, the place where watermark can be embedded is limited. Echo hiding has been mostly used in time domain.

As one of the most popular audio watermarking methods, spread spectrum techniques are originally used in radio communication to prevent the message interception. The basic idea of spread spectrum is to spread the watermark data across the entire audible spectrum. There are two main types of spread spectrum, which are Direct Sequence Spread Spectrum (DSSS) and Frequency Hopped Spread Spectrum (FHSS). The two major spread spectrum methods differ mainly in the way the encode the data

with the PN (Pseudo Noise) sequence. Cox et al. [28] first introduced to use spread spectrum method into watermarking system. In their proposed method, the original audio was modulated with the watermark and a PN sequence. As a result, the frequency spectrum of the data was spread over the available band, and the PN sequence appeared as random noise in the frequency domain. The final output signal is attenuated and added back to original audio signal. One big problem with spread spectrum systems is synchronization problem. In FSSS, it usually needs to cost great computational complexity to act against synchronization attacks. In DSSS, if perfect synchronization is not achieved, it usually uses some types of control systems to deal with synchronization problem. However, in a real-world situation, where perfect synchronization can never be guaranteed. Overall, the DSSS seems to be better if any type of control can be exerted over the given design area, but FSSS does have an advantage when no control can be exercised over the range and number of transmitters parameters. However, since most system designs involve some types of control over the environment that the communications system will be operated in, DSSS is currently the most common spread spectrum technique [29].

Ko et al. [30] presented a time-spread echo method as an alternative to the conventional echo in the echo hiding. Spreading an echo is performed in the impulse response domain using PN sequences. The proposed method is to combine echo hiding and spread spectrum technique benefits to achieve better imperceptibility of distortion. The authors evaluate the characteristics of the proposed method as a function of two parameters: amplitude and length of PN sequences used in spreading an echo. The PN sequences was used to make the amplitude of each component echo. The echo can be adjusted smaller and more accurate according to adjust the length of the PN sequences. Therefore, the time-spread echo method is easy to be implemented and enhances the

quality of the embedding.

Nowadays, masking is a technique widely used. It is to find out masking threshold to embed watermark into original audio signal and make it inaudible using a characteristic of the human audio system (HAS). The masking effect depends on the spectral and temporal characteristics of both the masked signal and the masker. Frequency masking refers to masking between frequency components in the audio signal and makes the simultaneous weaker signal inaudible. The masking threshold of a masker depends on the frequency, sound pressure level (SPL), and tone-like or noise-like characteristics of both the masker and the masked signal [18]. Temporal masking refers to both pre-masking and post-masking. Pre-masking affects weaker signals inaudible before the stronger masker is turned on, and post-masking affects weaker signals inaudible after the stronger masker is turned off [18]. Therefore, masking threshold can be defined by tonal components in frequency masking and pre-masking and post-masking in temporal masking. Temporal masking technique is used in MPEG-1 psychoacoustic model I. In the masking threshold calculation audible signals can be distinguished from those inaudible signals. As the inaudible signals may be compressed later, watermark should not be placed in those inaudible signals. Therefore the space where watermark can be embedded is limited. Fabien et al. [31] implemented Boney's temporal masking model on MPEG-1 layer 1 by using MATLAB.

Wavelet techniques are popularly used in audio watermarking methods in recent years. Wavelet transforms are considered forms of time-frequency representation and so are related to harmonic analysis. Almost all practically useful discrete wavelet transforms use filterbanks. These filterbanks may contain either finite impulse response (FIR) or infinite impulse response (IIR) filters. Wavelet transforms are broadly divided into three classes: continuous, discrete and multiresolution-based [32]. Xiang [33] pro-

posed a DWT (Discrete Wavelet Transform) based watermarking method against both TSM (time-scale modification) and MP3 compression. The Discrete Wavelet Transform (DWT), which is based on subband coding is found to yield a fast computation of Wavelet Transform. It is easy to implement and reduce the computation time and resources required [34]. In the case of DWT, a time-scale representation of the digital signal is obtained using digital filtering techniques. The signal to be analyzed is passed through filters with different cutoff frequencies at different scales. Therefore, the DWT is computed by successive lowpass and highpass filtering of the discrete time-domain signal, which is a good characteristic to deal with TSM attacks. In DWT, the most prominent information in the signal appears in high amplitudes and the less prominent information appears in very low amplitudes. Data compression can be achieved by discarding these low amplitudes. The wavelet transforms enables high compression ratios with good quality of reconstruction [34]. Therefore, DWT algorithms perform well against compression attacks.

Tu et al. [27] proposed a semi-fragile digital audio watermarking scheme which can be applied to content authentication and copyright verification for audio signal. The watermark is embedded into discrete wavelet domain by quantizing the wavelet coefficients. Wavelet domain has both spatial and frequency information simultaneously. When watermark embedding, in wavelet domain, the frequency information is used to embed watermark robust against attacks and the spatial information is used for content authentication. Quantization was also used to extract the watermark out without original audio signal. During watermark detection, a classical matching filter is employed to locate the beginning of the original audio accurately since some attacks may change the length of the audio. The quantization parameter is experiment-based.

Cai et al. [35] proposed an objective speech quality evaluation method using digital

audio watermarking. This method is based on the Digital Wavelet Transform (DWT) and quantization. Quantization was used to embed and extract the watermark. The selection of quantization parameter affects the efficiency and accuracy of the proposed method. Perceptual Evaluation of Speech Quality Mean Opinion Score (PESQMOS) was used to evaluate the speech quality. Using PESQMOS as a reference, the experimental results show that this method gives accurate predictions of subjective quality for speech signals.

Li et al. [36] proposed an algorithm of audio watermarking based on SNR to determine a scaling parameter (α). A scheme of audio digital watermark embedding and detection based on DWT is proposed in this paper. The intensity of embedded watermarks on the different audio segments were modified by adaptively modulating the scaling parameter (α). When watermark embedding, the authors set the SNR goal first and then adjust scaling parameter (α) and wavelet coefficients according to wavelet domain level and position decomposition. This way can avoid lots of experiments to adjust scaling parameter alpha (α) to achieve a good watermarking performance. While watermark detection, apply DWT to each segment of audio signal and correlation coefficient was adopted to calculate the difference of exacted watermark and original watermark. Correlation degree was calculated to determine the possibility of the watermark existence and then extract watermark out. The algorithm is robust to many attacks but it is lack of self-adaptability. If SNR is not used, the scheme will be short of self-adaptability and thus will not have the desired robustness against low pass filtering, additive noise, removing noise, and others.

Kaabneh et al. [37] proposed a watermark embedding method during audio signal mute period. Audio signal mute periods are the special periods in an audio signal. When watermark embedding, the proposed method searched all the mute periods in

an audio signal that falls within a certain predetermined threshold and then embedded watermark into those mute periods. When detection, all mute periods needed to be detected. The advantages of this method was a mute period becomes an integral part of any audio signal. It cannot be omitted since it represented an integral part of the audio signal. It occurred randomly in an audio signal, which was generated by the music producing process. And, a mute period represents a real time interval that would not be decreased when compressed. However, this method may not secure and cannot survive attacks, such as chopping, low pass filtering, re-sampling and re-quantization.

Cheng et al. [38] proposed an enhanced spread spectrum watermarking method for MPEG-2 AAC (Advanced Audio Coding) with low structural and computational complexity. The proposed algorithm aims at the host signal into one with smaller variance before adding the watermark. When watermark embedding, the algorithm sorts the original sequence in ascending order and constructs a new host signal by taking the difference of every pair of two consecutive samples in the sorted sequence and then add watermark into the new host signal by modifying the sorted coefficients differences. To decode, the sorting indices and watermark key are required to extract watermark out. This watermarking method works on AAC audio directly and low computational complexity, making it suitable for real-time watermarking application.

Tachibana [39] proposed to use a two-dimensional pseudo random array as watermarking identification key to embed and extract watermarks in MPEG-2 Advanced Audio Coding (AAC) bitstream. Detection is done by correlating the magnitudes of the frame contents with the pseudo-random array. According to the conclusion of the this paper, the robustness of the method is weaker than uncompressed-domain based watermarking and the acoustic quality has not been given.

Ni et al. [40] presented an adaptive additive watermarking model and analyzed cor-

responding detection performance. This paper constructed a general model to describe the adaptive watermarking system. In this paper, it summarized three main parts to design an adaptive watermarking system. The first is to design the watermark signal, the second is to design the embedding algorithm and the third is to design the extraction algorithm. Statistical algorithms were suggested to use in the adaptive watermarking algorithm. Meanwhile, the detection performance is analyzed in detail. Two kinds of false possibility, the possibility of false positive and false negative are considered during detection performance analysis. But some attacks such as MP3 compression, low-pass filter or Gaussian noise attacks are not considered in this paper.

Wu et al. [41] proposed a self-adaptive video watermarking algorithm. They introduced the statistical analysis for adaptive watermark embedding, but the watermark detection was not blind.

In the literature, quite a number of audio watermarking algorithms have been proposed. However, only a few of them directly embed watermark in the MP3 stream to provide superior performance against MP3 compression. MP3 watermarking methods will be reviewed in the next section.

3.6 Literature Review on MP3 Audio Watermarking

Early to 2000, Thorwirth [42] raised MP3 illegal copyright issue. In this paper, it compared encryption algorithms and proposed to use watermarking method to protect MP3 copyright. Watermark is added to the audio stream in order to protect individual layers of audio quality. Watermarking technology is the technique to embed watermark into original signal directly, therefore, it can be a complete solution in network-based

secure audio delivery and copyright protection.

Wang et al. [43] proposed a watermarking algorithm directly on MP3 signal. In this paper, the authors choose to embed watermark during MDCT transformation when performing the compression procedure. When watermark embedding, the authors choose to embed watermark into low frequency for robustness. The position of the MDCT coefficients recorded watermark data sequence. The position of the first non-zero is the beginning of the watermark data. MDCT coefficients were calculated and adjusted as the parameter of the watermark during MDCT procedure. After dequantization, IMDCT values are analyzed to obtain the watermark coefficients, and watermark is then detected during the IMDCT procedure. The position of the first non-zero is the start position of the watermark extraction. This watermarking method works on MP3 directly but it is not an self-adaptive watermarking method and the watermark space is limited to low frequency region.

The following papers are not on MP3 watermarking, however they report the researches on MP3 technology. For example, a standard quantization in MP3 was two-nested loop, one is inner loop to adjust global gain and the other one is outer loop to adjust scalefactor bands. Once the global gain is changed, the distortion of each scalefactor band is also changed. Therefore it is a two-loops computation, which consume lots time and cannot apply on real-time application. To solve the problem, You et al. [13] presented to use a look up table to make a single three-iteration loop algorithm instead of standard two-nested loop algorithm to reduce the computational burden.

Nikolajevic et al. [44] proposed an improved algorithm for fast implementation of MDCT and IMDCT (inverse MDCT) for MPEG-1 layer 3 and keep the audio quality. MP3 standard has adopted two MDCT block sizes to code associated audio signals: a short MDCT block (block size N = 12) and a long MDCT block (block size is N =

36). Since the length of MDCT (N) is not equal to 2^n, the MDCT computation cannot be applied efficiently. The proposed solution requires the size of the biggest block to be $N/2$. This block step modification allows efficient implementation of MDCT and IMDCT, especially on DSP. The proposed algorithm modified the standard ISO MP3 MDCT and IMDCT algorithm to achieve more effective performance.

Bohme et al. [45] surveyed the discipline to identify the available MP3 encoders and generated a data pool for analysis. The authors investigated and presented the analysis on the MP3 encoders and statistically analyzed MP3 data to enable detection of embedded information in MP3 files and streaming data. In their paper, LAME encoder and decoder open source code was introduced as the most professional MP3 encoder and decoder tool in both industry and academy.

Compared to the existed watermarking methods, our proposed new watermarking algorithms work on MP3 frames directly. During the watermark embedding and detection, the standard (ISO/IEC 11172-1:1993 Standard Part 3: Audio standard) MP3 encoder and decoder procedures are not changed. To develop an adaptive watermarking algorithm on MP3 with good performance during watermark embedding and to implement blind watermark detection are the major achievements in this study. The next two chapters will present the proposed watermarking algorithms in detail with experimental results to show that the new algorithms are robust against MP3 compression and most common signal manipulation and malicious attacks.

Chapter 4

The Proposed Adaptive Audio Watermarking Algorithms

In this chapter we will propose two adaptive watermarking algorithms for MPEG-1 Layer 3 (MP3) compressed audio for copyright protection.

In the first algorithm, Gaussian distribution analysis is used in both the watermark embedding and watermark detection processes. In watermark embedding, Gaussian distribution analysis values and energy calculation values on original audio signal are used to control the embedding strength. In watermark detection, Gaussian distribution analysis is used to determine watermark existence so that watermark can be extracted out appropriately. This Gaussian distribution based watermarking algorithm has shown good performance on MP3 compression.

However, the first algorithm has weakness when used in applications in which security is required because an attacker also can detect the existence of watermark by using Gaussian distribution analysis. To provide security, a correlation based watermarking algorithm is also proposed in this chapter.

Though this second algorithm has enhanced security on watermark data, our experiments show that the first algorithm shows better performance under the MP3 compression attacks. Therefore, the two algorithms can be used in different applications.

Both the algorithms use MP3 psychoacoustic model II on MP3 frames to control watermarking parameters. Both algorithms support blind watermark detection.

4.1 The Diagram of the Watermark Embedding Algorithms

Both the Gaussian distribution based watermarking algorithm and the correlation based watermarking algorithm embed watermark in MP3 frames directly. The key idea of the two watermarking algorithms is to adaptively adjust the watermarking coefficients according to the strength of original audio signal and human auditory system. Figure 4.1 presents the diagram of the watermark embedding algorithms.

4.2 Embedding Position

The original audio signals we used are in PCM (Pulse-Code Modulation) format and saved as .wav files. After the standard Fast Fourier Transform (FFT) and the filterbank in MP3, the original audio is divided into frames in frequency domain. The two proposed watermarking algorithms choose to embed watermark after Modified Modified Discrete Cosine Transform (MDCT) and before quantization.

The primary reason to choose embedding watermark after MDCT transformation and before quantization is for audio quality consideration. During the MP3 encoding process, two-time compressions happen. One is the hybrid filterbank with psychoacous-

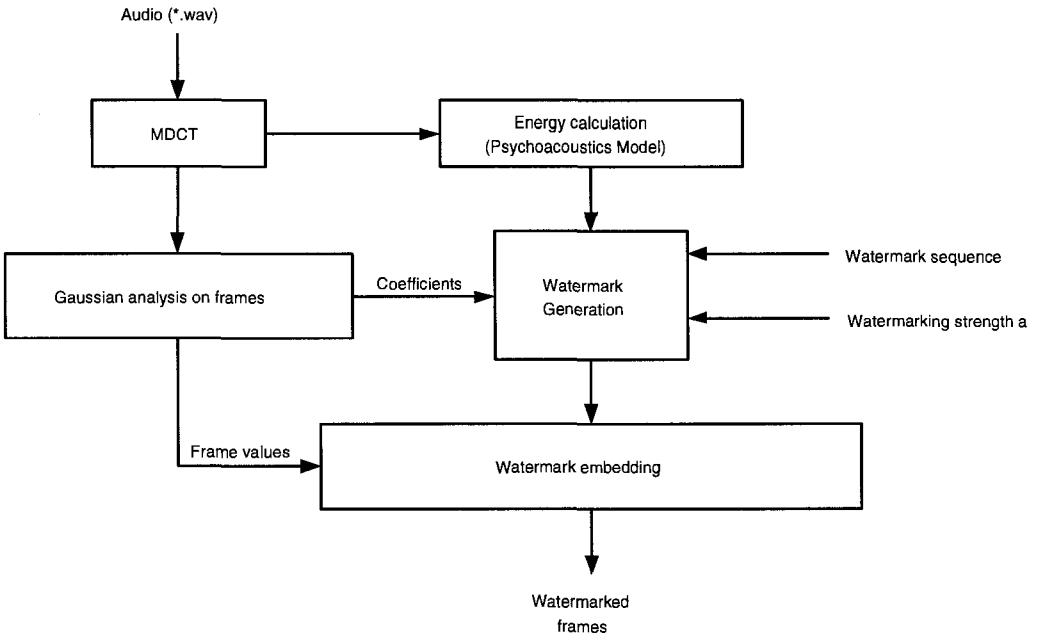

Figure 4.1: Diagram of watermark embedding algorithms.

tic model II masking threshold mechanism and the other one is the quantization. By applying the MDCT to each frame of subband samples, the 32 subbands are split into 18 frequency lines, creating a granule with total of 576 (32×18) lines of data. Prior to the MDCT, each subband signal has to be windowed. The psychoacoustic model II determines which window type to apply. If the psychoacoustic model decides that the subband signal at the present time frame shows little difference from the pervious time frame, then the long window type is applied, which will enhance the spectral resolution given by MDCT; if the psychoacoustic model decides that the subband signal shows considerable difference from the pervious time frame, then the short windows is applied. This window consists of three short overlapped windows and improves the time resolution given by the MDCT [15].

On the other hand, the quantization in MP3 is a non-uniform quantization mapping amplitude values into finite number of bits. There are two-nested loop during the

quantization. One is inner iteration loop to control the quantization step size and the other one is outer iteration loop to control the noise shaping factors for each scalefactor band. Once the the quantization step size is changed, the outer iteration loop noise shaping factors for each scalefactor band is also changed. Therefore, the quantization process is a two-loops adjustment and an unstable step for embedding watermark. Thus we choose to embed watermark before quantization. The following equations can illustrate the major steps in MP3 compression.

$$X_{mp3} = f_{Huffman}(f_{Quantization}(f_{MDCT}(X_{wav}))) \tag{4.1}$$

Watermark can be embedded in row data according to Equ. (4.2), after MDCT according to Equ. (4.3), or after quantization according to Equ. (4.4).

$$X'_{mp3} = f_{Huffman}(f_{Quantization}(f_{MDCT}(X_{wav} + W))) \tag{4.2}$$

$$X''_{mp3} = f_{Huffman}(f_{Quantization}(f_{MDCT}(X_{wav}) + W)) \tag{4.3}$$

$$X'''_{mp3} = f_{Huffman}(f_{Quantization}(f_{MDCT}(X_{wav})) + W) \tag{4.4}$$

W is the watermark; X_{wav} represents the original audio signal; f_{MDCT} is the MDCT process; $f_{Huffman}$ is the Huffman coding process and $f_{Quantization}$ represents the quantization process.

We aim at a watermarking algorithm that performs particularly well for MP3 compression in terms of inaudibility and robustness. Watermark should be embedded with maximum-possible energy (robustness) by considering human auditory system. There-

fore, we decide to embed watermark after MDCT and before quantization.

4.3 Watermark Data

We generate a 64-bit PN sequence (also called watermark sequence in this thesis), $W = \{w_i : i = 1, 2, \cdots, 64\}$, belonging to {0, 1}. Then, a dictionary is used to code each 1-bit PN number (also called watermark bit in this thesis) into more than one bit. Users can define their own dictionary data set to represent an actual 1-bit PN number. In this thesis, we set our dictionary rate N=3. Therefore, the length of the coded PN sequence (watermark sequence) will be 64×3 and the coded PN sequence can be represented as $W_1 = \{w_{1j} : j = 1, 2, \cdots, 192\}$, where W_1 is the watermark data after dictionary translation.

Table 4.1: Coding dictionary

Watermark bit	Representation
0	{0, 1, 0}
1	{1, 0, 1}

This approach allows users to define their own watermark package. Users can design their {0, 1} set or the number for one-bit watermark to represent one-bit PN number. Therefore, this method allows the length of watermarks to be flexible. Refer to Table 4.1, in this thesis, {0, 1, 0} is used to represent one watermark bit "0" and {1, 0, 1} is used to represent one watermark bit "1". Meanwhile, this dictionary can also work against attacks. Attackers can attack the embedded $\{0, 1\}$ watermark sequence, which is not the embedded message. To know the final message, one needs to look up the dictionary for which combination represents watermark bit "0" and watermark bit "1". And only the watermark owner knows the dictionary.

4.4 Structure of Watermarking Frame

Frames forms MP3 audio stream after the MDCT. According to the experimental results, if two (or more) original frames (32×18 matrix) are merged to a macro frame (32×36 matrix), the MDCT values in the macro frame follow the Gaussian distribution. One original frame alone can also be used to carry watermark, but the performance is not as good as when two frames are merged.

It was also found that values in a macro frame formed by merging more original frames follow the Gaussian distribution more closely. However, after many experiments, it is decided that merging two frames together will generate good result while not wasting too much watermarking space.

When an audio signal is stereo, for the simplicity of implementation, only one channel (right channel) is used even though there are two channels. So, actually the implementation is on mono audio (single channel) and the granule is a frame. Figure 4.2 presents the MDCT value distribution after two original frames are merged to form one macro watermarking frame unit. In Figure 4.2, the x-axis represents the MDCT values in a macro frame, and the y-axis represents the number of the MDCT values in that macro frame.

In this thesis, two original frames are merged to form one macro frame, and one bit ("0" or "1") is embedded in one macro frame. This means that one bit "0" or "1" is embedded in two original frames as shown in Figure 4.3.

An MP3 frame after MDCT is a 32×18 (32 subbands, each with 18 frequency lines) coefficient matrix. Each bit of our coded watermark sequence spreads into an macro frame composed of two original MP3 frames. The spreading factor is equal to the size of an macro frame. That means, "0" will be represents by 1152 "−1", and "1" will be

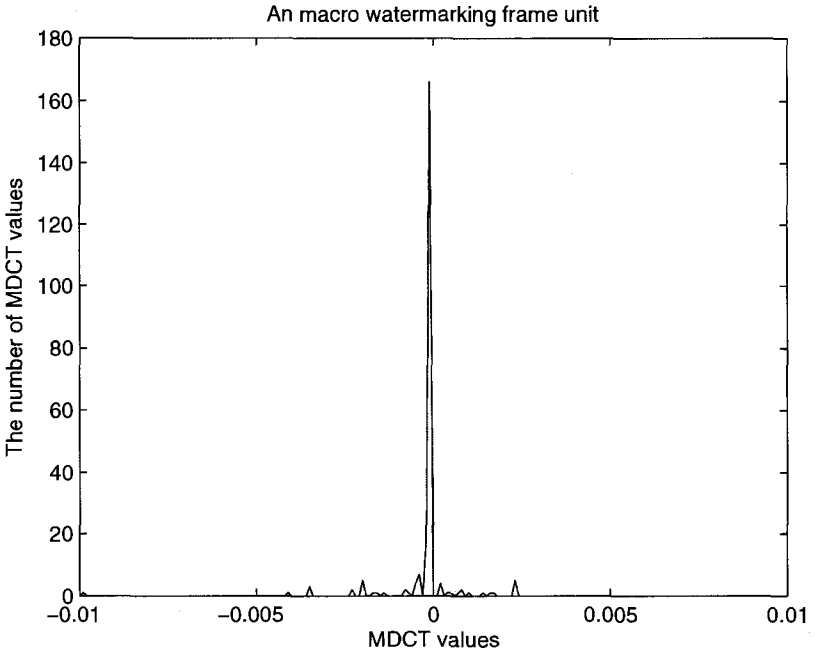

Figure 4.2: The distribution of MDCT values in a macro frame.

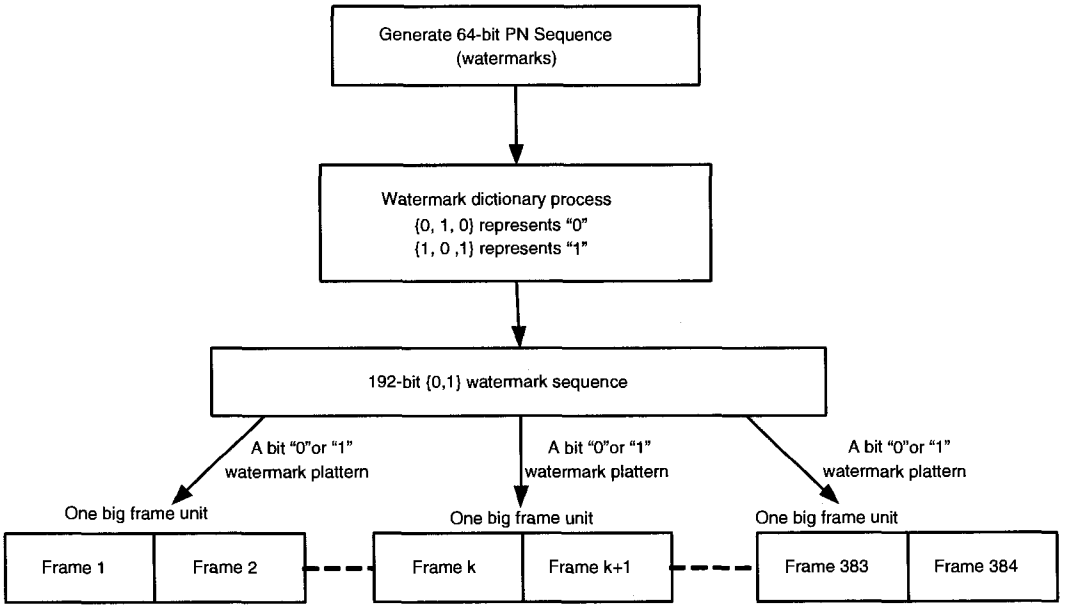

Figure 4.3: The proposed watermarking frame structure.

represented by 1152 "+1". The 1152 consecutive "−1" or "+1" will be added to the 1152 MDCT coefficients in an macro frame, which can be illustrated by Equ. (4.5).

$$v'_k = v_k + \alpha_k w_{3k} \tag{4.5}$$

where v_k is the original MDCT coefficient, v'_k is the watermarked MDCT coefficient, w_{3k} is the bit to embed, and α_k is the embedding strength.

4.5 Energy Calculation for Adaptive Watermarking Coefficients

MP3 audio energy is calculated by using the psychoacoustics model II defined in ISO/IEC 11172, Part 3: Audio. The energy calculations in this model are shown in Figure 4.4 [1] [2], and described as follows.

1. Reconstruct 1024 samples of the input signal.

   Generate audio samples for an PCM audio stream input. The samples are made available at every call to the threshold generator. The threshold generator must store 1024 samples, and concatenate those samples to accurately reconstruct 1024 consecutive samples of the input signal.

2. 1024-point FFT.

   1024-point FFT for audio spectral lines of the input signal, $s(i)$, where i represents the index, $1 < i < 1024$ of the current input stream.

3. Calculate the complex spectrum of the input signal.

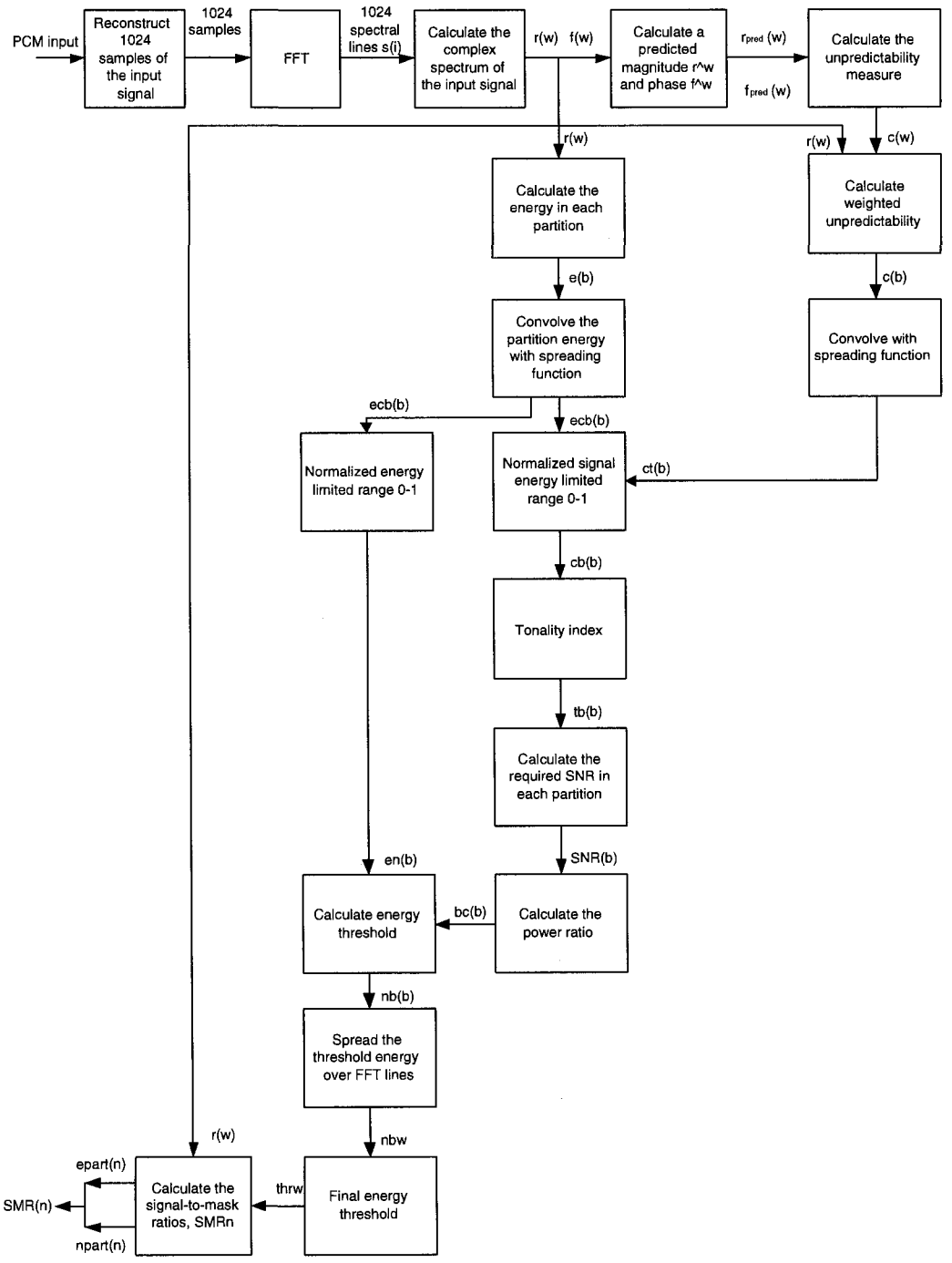

Figure 4.4: Steps for threshold calculation. [1] [2]

First, $s(i)$ is windowed by a 1024 point Hann window. Second, a standard forward FFT of $s_w(i)$ is calculated, where w represents the window frequency. Third, the polar representation of the transform is calculated. $r(w)$ and $f(w)$ represent the magnitude and phase components of the transformed $s_w(i)$, respectively.

4. Calculate a predicted magnitude and phase.

   The predicted magnitude, $r_{pred}(w)$, and phase, $f_{pred}(w)$ are calculated from the preceding two threshold calculation blocks r and f.

$$r_{pred}(w) = 2.0 * r(t-1) - r(t-2) \tag{4.6}$$

$$f_{pred}(w) = 2.0 * f(t-1) - f(t-2) \tag{4.7}$$

   where t represents the current block number, $t-1$ indexes the previous block's data, and $t-2$ indexes the data from the threshold calculation block before that.

5. Calculate the unpredictability measure $c(w)$.

$$temp1(w) = r(w) * \cos(f(w)) - r_{pred}(w) * \cos(r_{pred}(w)) \tag{4.8}$$

$$temp2(w) = r(w) * \sin(f(w)) - f_{pred}(w) * \sin(f_{pred}(w)) \tag{4.9}$$

$$c(w) = \frac{\sqrt{[temp1(w)]^2 + [temp2(w)]^2}}{r(w) + \mid r_{pred}(w) \mid} \tag{4.10}$$

If signal is a tonality, the value of $c(w)$ usually is small. In general, if $c(w) = 0$, we see the signal as a tonality. Otherwise, if $c(w) = 1$, we see it as a noise signal.

6. Calculate the energy and unpredictability in the threshold calculation partitions.

   The energy in each partition, the partition energy $e(b)$, where, b is a partition band, is:

$$e(b) = \sum_{w=w_{low}(b)}^{w_{high}(b)} r(w)^2 \tag{4.11}$$

   and the weighted unpredictability, $c(b)$, is:

$$c(b) = \sum_{w=w_{low}(b)}^{w_{high}(b)} r(w)^2 c(w) \tag{4.12}$$

   where, the index of the calculation partition is b; the lowest frequency line in the partition is $w_{low}(b)$; and the highest frequency line in the partition is $w_{high}(b)$.

7. Convolve the partitioned energy and unpredictability with the spreading function.

   Several points in the following description refer to the "spreading function". It is calculated by the following equations:

   Assume that i is the bark value of the signal being spread, j is the bark value of the band being spread into, and $tmpx$ is a temporary variable.

   If $j \geq i$,

$$tmpx = 3.0 * (j - i) \tag{4.13}$$

   else,

$$tmpx = 1.5 * (j - i) \tag{4.14}$$

$$tmpz = 8 * minimum((tmpx - 0.5)^2 - 2 * (tmpx - 0.5), 0) \tag{4.15}$$

where, *tmpz* is a temporary variable, and minimum (a, b) is a function returning the more negative value of a or b.

$$tmpy = 15.811389 + 7.5 * (tmpx + 0.474) - 17.5 * (1.0 + (tmpx + 0.474)^2)^{0.5} \tag{4.16}$$

where, *tmpy* is another temporary variable.

If $tmpy < -100$ then $sprdngf(i, j) = 0$; else, $sprdngf(i, j) = 10^{\frac{(tmpz+tmpy)}{10}}$.

Therefore, for each partition b, the partitioned energy and unpredictability with the spreading function are:

$$ecb(b) = \sum_{bb=1}^{b_{max}} e(bb) * sprdngf(b_{val}(bb), b_{val}(b)) \tag{4.17}$$

$$ct(b) = \sum_{bb=1}^{b_{max}} c(bb) * sprdngf(b_{val}(bb), b_{val}(b)) \tag{4.18}$$

where, b_{max} is a largest value of b, equal to the largest index, exists for each sampling rate. The median bark value of the partition b is $b_{val}(b)$.

Because $ct(b)$ is weighted by the signal energy, it must be renormalized to $cb(b)$.

$$cb(b) = ct(b)/ecb(b) \tag{4.19}$$

At the same time, due to the non-normalized nature of the spreading function, $ecb(b)$ should be renormalized and the normalized energy $en(b)$ is calculated as:

$$en(b) = ecb(b) * r_{norm}(b) \tag{4.20}$$

where, the normalization coefficient, $r_{norm}(b)$ is:

$$r_{norm}(b) = 1/(\sum_{bb=0}^{b_{max}} sprdngf(b_{val}(bb), b_{val}(b))) \tag{4.21}$$

8. Convert $cb(b)$ to $tb(b)$, the tonality index.

$$tb(b) = -0.299 - 0.43 * log_e cb(b) \tag{4.22}$$

Each $tb(b)$ is limited to the range of 0 and 1.

9. Calculate the required SNR in each partition

NMT(b) = 5.5 dB for all b, where, $NMT(b)$ is the value for noise masking tone (in dB) for the partition. TMN(b) = 18 dB for all b, where, $TMN(b)$ is the value for tone masking noise (in dB) for the partition. The required signal to noise ratio, $SNR(b)$, is:

$$SNR(b) = tb(b) * TMN(b) + (1 - tb(b)) * NMT(b) \tag{4.23}$$

10. Calculate the power ratio.

    For each partition b, we calculate its power ratio $bc(b)$ through $SNR(b)$:

$$bc(b) = 10^{-SNR(b)/10} \tag{4.24}$$

11. Calculation of actual energy threshold, $nb(b)$, in each partition b.

$$nb(b) = en(b) * bc(b) \tag{4.25}$$

12. Spread the threshold energy over FFT lines, yielding nbw.

$$nbw = nb(b)/(w_{high}(b) - w_{low}(b) + 1) \tag{4.26}$$

    where, $w_{high}(b)$ is the highest threshold of the partition b, and $w_{low}(b)$ is the lowest threshold of the partition b.

13. Include absolute thresholds $absthrw$, yielding the final energy threshold of audibility, $thrw$.

$$thrw = max(nbw, absthrw) \tag{4.27}$$

    The dB values must be converted into the energy domain after considering the FFT normalization actually used.

14. Calculate the signal-to-mask ratios, $SMR(n)$.

    The energy in the scalefactor band, $epart(n)$, is:

$$epart(n) = \sum_{w=w_{low}(n)}^{w_{high}(n)} r(w)^2 \tag{4.28}$$

where, n is the index of the coder partition.

Then, if $width(n) = 1$, the noise level in the scalefactor band, $npart(n)$ is calculated as:

$$npart(n) = \sum_{w=w_{low}(n)}^{w_{high}(n)} thrw \tag{4.29}$$

else,

$$npart(n) = minimum(thr_{w_{low}(n)}, \cdots, thr_{w_{high}(n)}) * (w_{high}(n) - w_{low}(n) + 1) \tag{4.30}$$

where, $w_{low}(n)$ is the lower index of the coder partition (n) , $w_{high}(n)$ is the upper index of the coder partition (n). minimum (a, $\cdots$, z) is a function returning the smallest positive value of the arguments a, $\cdots$, z.

The ratios to be sent to the coder, $SMR(n)$, are calculated as:

$$SMR(n) = 10log_{10}(epart(n)/npart(n)) \tag{4.31}$$

The psychoacoustic model determines the window type on the frame based on the calculated masking threshold SMR(n) value. The calculated energy will be used later in MP3 MDCT calculations. However, the energy calculation is based on partition and MP3 MDCT calculation is based on subbands, and the partitions do not have one to

one correspondence to subbands. The following explains how the energy values from the psychoacoustic model partitions are mapped to the MDCT values in subbands. The Scalefactor bands behave as the bridge between the partition and subbands. ISO/IEC 11172, Part 3: Auido, Annex C defines the energy value mapping from the partition to scalefactor band; and IOS/IEC 11172, Part3: Audio Table B further defines the energy calculations on scalefactor bands to the MDCT subband values. Figure 4.5 [1] illustrates the relationship among MDCT value, scalefactor band and energy.

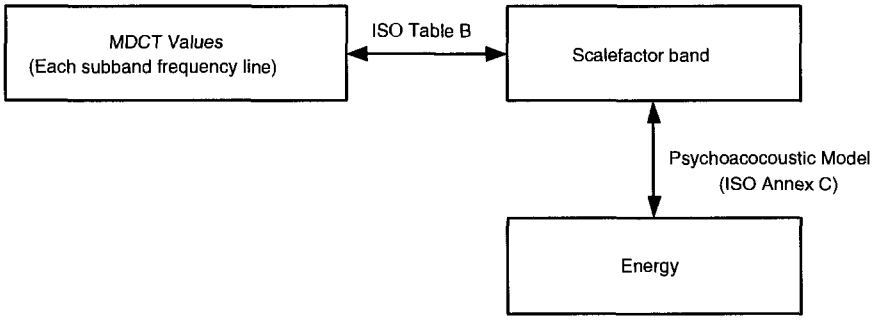

Figure 4.5: The relationship between MDCT and energy.

As an example, Table 4.2 shows the mapping between the partitions and scalefactor bands. In Table 4.2, the sampling frequency is 32 kHz and long block is used. "no. sb" stands for the index number of a scalefactor band; "cbw" stands for the number of partitions converted to one scalefactor band (excluding the first and the last partition); "bo" and "bu" are the first and last partition that is mapped to the scaleband respectively; "w1" and "w2" are the beginning weight and ending weight respectively. For example, in the second line of Table 4.2, Partition 2 and Partition 3 are mapped to scalefactor band 1 with the starting weight 0.472 and ending weight 0.305.

In MP3 both long block and short block may be used. For long block, there are 21 scalefactor bands with one energy value for each band; for short block, while there are 12 scalefactor bands with one energy value for each band. The mapping between the

Table 4.2: Energy values mapping between partition and scalefactor bands (sampling rate = 32 kHz, long block)

no. sb	cbw	bu	bo	w1	w2
0	1	0	2	1.000	0.528
1	2	2	4	0.472	0.305
2	2	4	6	0.694	0.083
3	1	6	7	0.917	0.861
4	2	7	9	0.139	0.639
5	2	9	11	0.361	0.417
6	3	11	14	0.583	0.083
7	2	14	16	0.917	0.750
8	3	16	19	0.250	0.870
9	3	19	22	0.130	0.833
10	4	22	26	0.167	0.389
11	4	26	30	0.611	0.478
12	4	30	34	0.522	0.033
13	3	34	37	0.967	0.917
14	4	37	41	0.083	0.617
15	3	41	44	0.383	0.995
16	4	44	48	0.005	0.274
17	3	48	51	0.726	0.480
18	3	51	54	0.519	0.261
19	2	54	56	0.739	0.884
20	2	56	58	0.116	1.000

partitions and scalefactor bands is similar to that shown in Table 4.2.

The mapping between the scalefactor bands to the MDCT subbands is affected by frequencies. In low frequencies, several scalefactor bands can be mapped to the same MDCT subband; in high frequencies, one scalefactor band can be mapped to multiple MDCT subbands. As an example, Table 4.3 shows the mapping between the scalefactor bands and MDCT subband when the sample frequency is 32 kHz and long block is used. In the table, "scalefactor band" is the index of the scalefactor band, "width of band" is the number of MDCT subband frequency lines, and "index of start" and "index of end" are the corresponding starting and ending frequency line indices. For example, in the second line of Table 4.3, scalefactor band 1 is mapped to 4 MDCT subband frequency lines: from frequency line 4 for frequency line 7. Note that there are totally 32*18=576 frequency lines. In case the frequency line number are bigger than the highest number in the table (i.e. > 549) the MDCT energy value will be filled with zero.

Table 4.3: Relationship between scalefactor bands and subbands (32 kHz sampling rate and long block)

scalefactor band	width of band	index of start	index of end
0	4	0	3
1	4	4	7
2	4	8	11
3	4	12	15
4	4	16	19
5	4	20	23
6	6	24	29
7	6	30	35
8	8	36	43
9	10	44	53
10	12	54	65
11	16	66	81
12	20	82	101
13	24	102	125
14	30	126	155
15	38	156	193
16	46	194	239
17	56	240	295
18	68	296	363
19	84	364	447
20	102	448	549

Combining Table 4.2 and Table 4.3 we obtain the mapping from the psychoacoustic model partitions to the MDCT subbands. For example, in the second line of Table 4.2, Partition 2 and Partition 3 are mapped to scalefactor band 1 with the starting weight 0.472 and ending weight 0.305. Then, in the second line of Table 4.3, scalefactor band 1 is further mapped to MDCT frequency line 4 through frequency line 7.

4.6 Watermarking Space

In our experiments we initially embedded watermark of the same energy in two different ways. One way is to embed watermark at certain positions of the original audio and the other is to spread watermark of the same energy into the original audio. Even though the SNR and PSNR results are the same, spreading watermark into the original audio makes watermarked audio better perceptual quality to human. We thus focus on the

second approach.

Spreading watermark also increases the watermark security. For example, for spreading watermark, an attacker who attempts to attack a watermarking algorithm will find it more difficult to locate the watermark position without destroying the whole audio signal.

Our watermark spreading is feasible because the watermarking coefficients will adjust adaptively according to the original audio strength. Instead of embedding watermark in just low frequency region or certain places, we can embed watermark in various frequencies or in all frequencies. The watermarking space is increased.

4.7 MP3 Compression Rates

MP3 compression rates impact on audio quality. For example, 32 kbps compression bit rate usually removes lots of audio information during the compression. The higher the compression bit rate, the better the audio quality.

When a PCM audio is input to an MP3 encoder, the FFT transfers the audio to the frequency domain. The HAS (Human Auditory System) then processes perceived sound in non-equal width subband called critical bands (approximately the same as scalefactor bands) in psychoacoustics model II. The psychoacoustics model analyzes sound in each scalefactor band independently and calculates energy and masking threshold on each scalefactor band. The bandwidth for the scalefactor band is determined by the masking threshold calculation, MP3 compression rate and compression window types. As the result, MP3 compression on audio signal with different rates will generate different scalefactor bands with different energy values. If high audio quality signal meets a strong MP3 compression rate attacks, such as 32 kHz, 48 kHz or 64 kHz, the original

signal bandwidth is changed and some information might lost. Therefore, MP3 compression attacks usually damage the audio quality. On the other hand, a weaker MP3 compress rate attack usually impacts audio signal less.

In the proposed watermarking algorithm, the most commonly used compression bit rates are used. They are the following: 32 kbps, 48 kbps, 64 kbps, 80 kbps, 128 kbps and 320 kbps. We experimented these compression bit rates on both Gaussian distribution based watermarking algorithm and the correlation coefficient based watermarking algorithm. We use SNR to indicate the audio quality of the watermarked audio signal and BER to indicate the robustness of the watermarking algorithm. The following scenarios have been tested. First of all, when the same compression rate is used in both watermark embedding and detection, our test results show BER=0 for both the Gaussian distribution based watermarking algorithm and the correlation coefficient based watermarking algorithm, which means both watermarking methods are robust. Meanwhile, our test results show the SNR value is over 20 dB, which means the sound quality is good for both watermarking methods under this scenario. Secondly, if high quality audio signal meets a strong MP3 compression, the audio quality will suffer. For example, if a watermark is embedded using 128 kbps compression rate into an original 45 second 44.1k-16bit-mono.wav audio and if the audio is then compressed by 32 kbps MP3 compression rate, the resulting BER value is high and the resulting SNR value is less than 20 dB. However, when the same audio signal meets a weak MP3 compression, such as 320 kbps, the resulting BER value is equal to 0 and the resulting SNR is over 20 dB, which shows a satisfactory robustness and audio quality performance. If a low quality audio signal, for which 32 bps, 48 kbps or 60 bps is used to embed watermark, meets a weak MP3 compression, such as 128 kbps, 160 kbps or 320 kbps, then BER is equal to 0 and SNR is over 20 dB, which indicates good robustness and quality. However,

the overall sound quality of the Gaussian distribution based watermarking algorithm is better than that of the correlation coefficient based watermarking algorithm. We will present the detailed experimental results and explanations in Chapter 5.

4.8 Gaussian Distribution based Watermarking Algorithm

Gaussian Distribution based watermarking algorithm embeds watermark in MP3 frames directly. The core idea of this watermarking algorithm is to adaptively adjust the watermarking coefficients according to the strength of original audio signal. To implement this idea, Gaussian distribution analysis is used in both the watermark embedding and detection process.

4.8.1 Gaussian Distribution

Gaussian distribution, also called the normal distribution, is an important family of continuous probability distributions, applicable in many fields [46]. Each member (value, x) of the family may be defined by two parameters, location and scale: the mean (average, μ) and standard deviation (variability, σ), respectively shown in Equ.(4.32).

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/(2\sigma^2)} \tag{4.32}$$

The standard normal distribution is a Gaussian distribution with a mean (μ) of zero and a variance (σ^2) of one. For example, Figure 4.6 shows a set of values ranging from -3 to 3, which follows the standard normal distribution.

The importance of Gaussian distribution as a model of quantitative phenomena in

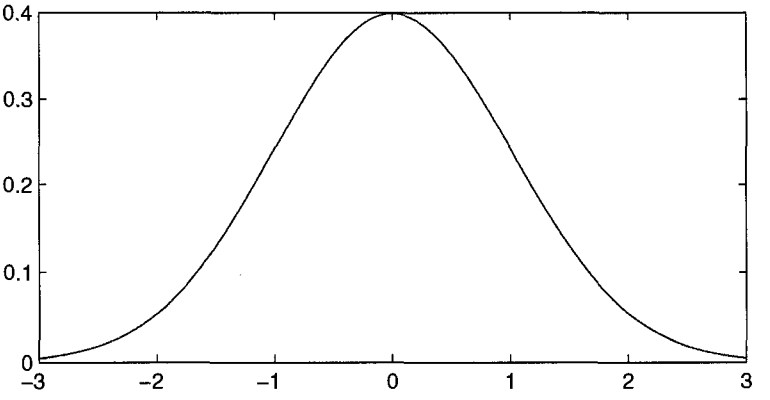

Figure 4.6: The standard normal distribution.

the natural and behavioral sciences is due to the central limit theorem. Many psychological measurements and physical phenomena (like noise) can be approximated well by the normal distribution [46].

Gaussian distribution also arises in many areas of statistics. For example, the sampling distribution of the sample mean is approximately normal, even if the distribution of the population from which the sample is taken is not normal.

In this study, it was found that the values in MP3 frames follow the standard normal distribution well, therefore Gaussian distribution analysis is employed in the proposed watermarking algorithm. When embedding, Gaussian distribution analysis is used to analyze original audio energy strength to determine the watermarking coefficients. Thus, the watermarked audio values in frame have skewness from the original audio mean. When detecting, Gaussian distribution is used to detect frame skewness (possibility that a frame contains a watermark) and to extract watermark out from the watermarked frames.

4.8.2 Watermark Embedding

The following gives detailed steps for watermark embedding.

1. Watermark sequence (64-bit PN sequence) is generated randomly. A dictionary-based method is used to spread watermark sequence to 192 bits.

2. Analyze the frames and determine the first available watermarking frame. For example, the first non-zero frame can be the candidate.

3. Merge two continuous frames (each frame with 32 subbands, and each subband with 18 frequency lines) to form one macro (32×36) frame for embedding watermark.

4. Generate the watermark pattern, (32×36) matrix, which is the same size as an macro frame. The content of the pattern is all "+1" or all "−1"to represent a bit of watermark "1" or a bit of watermark "0".

5. Generate Gaussian analysis values for all 32×36 MDCT values in the macro frame. Therefore, this step generates 1152 Gaussian analysis values. Equ. (4.33) shows the detailed Gaussian distribution calculation on each macro watermarking frame:

$$f(X_{ij}) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(X_{ij}-\overline{x})^2/(2\sigma^2)} \tag{4.33}$$

$$\overline{x} = \frac{1}{32 \times 36} \sum_{i=1}^{32} \sum_{j=1}^{36} X_{ij} \tag{4.34}$$

$$\sigma = \sqrt{\frac{1}{32 \times 36} \sum_{i=1}^{32} \sum_{j=1}^{36} (X_{ij} - \overline{x})^2} \tag{4.35}$$

where, X_{ij} (i= 1, 2, $\cdots$, 32, j= 1, 2, $\cdots$, 36) represents an MDCT value in a macro watermarking frame.

6. Calculate energy for scalefactor bands in MDCT.

7. Form the final watermark according to energy calculation, and Gaussian analysis results. Equ. (4.36) and Equ. (4.37) illustrate the watermark embedding.

$$X'_{ij} = X_{ij} + \Delta X_{ij} \tag{4.36}$$

where, X'_{ij} (i=1, 2, $\cdots$, 32 and j=1, 2, $\cdots$ 36) represents the watermarked MDCT values in a macro frame, and

$$\Delta X_{ij} = \alpha * \frac{log(energy_{ij})}{E} * f(X_{ij}) * W_{ij} \tag{4.37}$$

where, E =100, and α = 0.000001 is set based on trial and error. $f(X_{ij})$ is the Gaussian analysis value for (X_{ij}) in a frame, and $W_{ij}$is the watermark. If watermark bit "0" is embedded into a macro frame, the actual watermark W_{ij} is all "-1" representatively; If watermark bit "1" is embedded into a macro frame, the actual watermark W_{ij} is all "+1" representatively.

4.8.3 Comparison of Detection Techniques

To identify the best watermark detection method, three statistic methods have been implemented, which are respectively based on mean, skewness and Gaussian distribution.

1. Mean: For a data set $x_i = \{x_i : i = 1, 2, \cdots, n\}$, the mean, defined by Equ. (4.38) [47], is the sum of the observations divided by the number of observations.

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4.38}$$

For our implementation, n = 1152, which is the number of MDCT coefficients in an macro frame.

2. Skewness: In probability theory and statistics, skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable. For instance, a data set ($x_i = \{x_i : i = 1, 2, \cdots, n\}$) we can calculate skewness by using Equ. (4.39) [47].

$$Skewness = \frac{\sqrt{n(n-1)}}{n-2}\frac{\sqrt{n}\sum_{i=1}^{n}(x_i - \overline{x})^3}{(\sum_{i=1}^{n}(x_i - \overline{x})^2)^{3/2}} \tag{4.39}$$

where,

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4.40}$$

For our implementation, n = 1152, which is the number of MDCT coefficients in an macro frame.

3. Gaussian distribution analysis: Defined in Equ. (4.32)

Table 4.4 presents the bit-error-rate (BER) performance for watermark detection based on the above three statistic methods according to our experiments.

Due to its better performance, Gaussian distribution analysis is chosen for watermark detection in our implementation.

Table 4.4: Statistic methods for blind detection

Statistic method	Average BER
Mean	15.87 %
Skewness	14.94 %
Gaussian	0

4.8.4 Watermark Extraction

In this watermarking algorithm, Gaussian distribution analysis is used to detect the possibility of the watermark existence in a frame and make the blind watermark detection possible. When the analysis determines that the watermark is in the frame, we will try to extract watermark after alias detection and before IMDCT. The flow chart of the watermark extraction is illustrated by Figure 4.7.

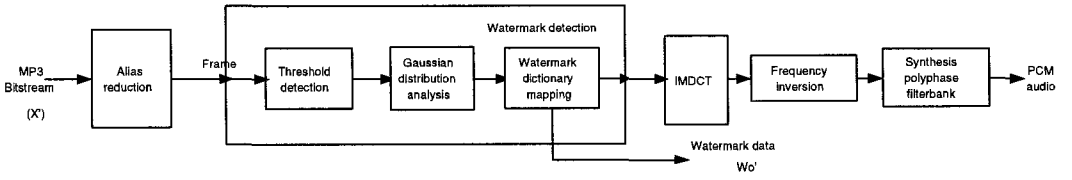

Figure 4.7: The proposed watermark extraction.

The following gives the detailed steps for the blind watermark extraction.

1. Find the first frame that contains watermark. According to the embedding algorithm, the first bit of watermark is embedded in the first non-zero frame. Therefore, the first non-zero frame should be the first detection candidate.

2. During embedding, two frames were merged to form one macro frame. One macro frame carries one bit watermark. Therefore, during the detection, two continuous frames are merged to one frame unit to detect.

3. Watermarked signals might have been changed due to various factors such as signal distortions, cropping, malicious attacks, signal compression, filtering or ran-

dom noise additions. An outlier is an observation that lies an abnormal distance from other values in a random sample from a population. The word "outlier" also appears in other places. In this thesis, signals are now quantified into data in the macro frames. Inside the macro frame (32×36 matrix) there may be some extremely large or extremely small values due to reasons such as noises during signal transmission. We consider those extreme values as abnormal and thus exclude them to prevent interference to the Gaussian analysis. The experiment-based outlier threshold $G_{max} = 100000.0$ and $G_{min} = 0.000001$ are use to define the signal outliers. Any values greater than $G_{max} = 100000.0$ or smaller than $G_{min} = 0.000001$, are treated as the signal outliers. Such signal outliers are then not used in the Gaussian analysis. Equ. (4.41) and Equ. (4.42) [48] show how to detect outlier data in each macro frame.

$$threshold = 3 * \sigma \tag{4.41}$$

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2} \tag{4.42}$$

For each macro frame, if the MDCT value belongs to outlier data, it will be seen as an unexpected value. The unexpected value will not be processed.

4. Apply Gaussian distribution analysis on all the MDCT values in the macro frame for watermark detection. Watermark W'_o is extracted based on Gaussian distribution analysis function D (X'_{ij}, σ) on watermarked macro frame.

$$W'_o = \sum_{i=1}^{32} \sum_{j=1}^{36} D(X'_{ij}, \sigma) \tag{4.43}$$

where, n=1152 is the number of the MDCT values in a macro frame, and

$$D(X'_{ij}, \sigma) = \left(\frac{1}{\sqrt{2\pi} * \sigma} \right) * e^{-\left[\frac{-\left[X'_{ij} - \sigma(X'_{ij})\right]}{n} \right]^{\frac{2}{2*\sigma^2}}} \tag{4.44}$$

where, the Gaussian distribution threshold $\sigma = 0.0005$ is determined based on experiments.

During detection, the sum W'_o of the Gaussian values is calculated in macro frame. If W'_o is greater than "+1", we determine that a watermark bit "1" was embedded into the original macro frame. If W'_o is less than "−1", we determine that a watermark bit "0" was embedded into the original macro frame. If W'_o is between $[-1, +1]$, we determine that no watermark was embedded into that macro frame.

In Figure 4.8, the middle curve shows the distribution of detection values resulted from original macro frame (without watermark). The left dashed curve shows the distribution of the detection values resulted from watermarked macro frame with watermark 0. The right dashed curve shows the distribution of the detection values resulted from watermarked macro frame with watermark 1.

5. Look up the dictionary table to extract watermark out.

4.9 Correlation based Watermarking Algorithm

For correlation based watermarking, the correlation coefficient is used to detect the probability of watermark existence and extract the watermark out when there exists a

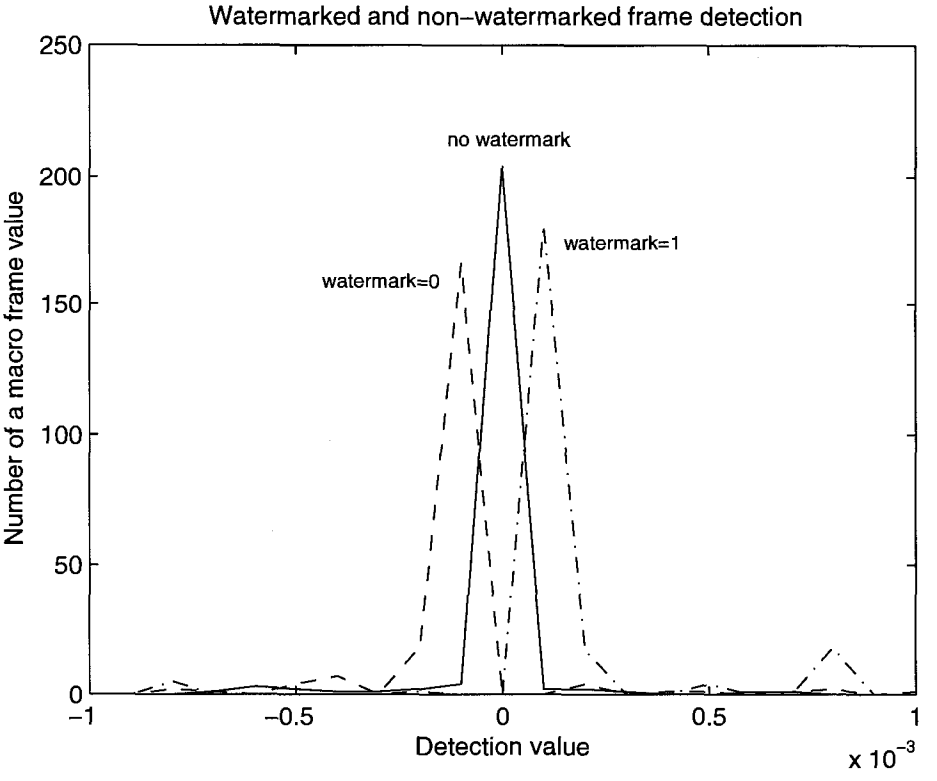

Figure 4.8: Detection results for watermarked and non-watermarked frame.

watermark.

4.9.1 Correlation Coefficient

Correlation coefficient indicates the strength and direction of a linear relationship between two random variables [49]. It varies from 0 (random relationship) to 1 (perfect linear relationship) or -1 (perfect negative linear relationship).

Correlation coefficients differ from linear correlations in two ways. First, we subtract the means of the two vectors before correlating them. Thus, the detection value is unaffected if a constant is added to all elements of either of the vectors. Second, we normalized the linear correlation by the magnitudes of the two vectors. Thus, the

detection value is unaffected if all elements of either vector are multiplied by a constant. As a result, the correlation coefficient is defined as [3]:

$$z_{cc}(v, w_r) = \frac{\widetilde{v} \cdot \widetilde{w}_r}{\sqrt{(\widetilde{v} \cdot \widetilde{v})(\widetilde{w}_r \cdot \widetilde{w}_r)}} \tag{4.45}$$

where $\widetilde{v} = (v - \overline{v})$, and $\widetilde{w}_r = (w - \overline{w}_r)$, $\overline{v}$ and $\overline{w}_r$ are the mean values of v and w_r, and

$$\widetilde{v} \cdot \widetilde{w}_r = \sum \widetilde{v}\widetilde{w}_r \tag{4.46}$$

The correlation coefficient can be interpreted as the inner product of $\widetilde{v}$ and $\widetilde{w}_r$, after each has been normalized to have unit magnitude. This is simply the cosine of the angle between the vectors and is bounded thus [3]:

$$-1 \leq z_{cc}(v, w_r) \leq 1 \tag{4.47}$$

The detector outputs is as [3]:

$$Mn= \begin{cases} 1, & \text{if } z_{cc}(v, w_r) > \tau_{cc} \\ no \ \ watermark & \text{if } -\tau_{cc} \leq z_{cc}(v, w_r) \leq \tau_{cc} \\ 0, & \text{if } z_{cc}(v, w_r) < -\tau_{cc} \end{cases}$$

where τ_{cc} is a constant threshold.

4.9.2 Watermark Embedding

In this algorithm, we still choose to embed watermark between MDCT process and quantization. Watermark dictionary idea is still used to make watermark data. Energy calculation is calculated on each MP3 frame as one of the watermark parameters for adjustment. However, in this method, in order to enhance watermark security, correlation coefficient is used on watermark random pattern. The following procedure gives the detailed watermark embedding steps.

1. Generate 64-bit watermark sequence and expand the sequence to 192 bits by dictionary mapping. Each bit of the expanded sequence will be embedded into one macro frame.

2. Determine the first suitable watermarking frame. For example, the first non-zero frame can be the candidate.

3. Merge two neighboring frames (each frame with 32 subbands, and each subband with 18 frequency lines) to form one macro frame (32×36) for watermark embedding.

4. Generate a random pattern, R_p, and calculate the correlation coefficient, z_{cc}, between MDCT values and random pattern.

5. Assign a sign to the random watermarking pattern in a macro frame. The first frame in a macro frame is assigned $+1$, and the second frame is assigned -1. Figure 4.9 illustrates the random watermarking pattern sign.

   Watermark bit after the dictionary expansion is to generate a dictionary based watermark. For example, we define 3-bit sequence to represent one bit binary. a

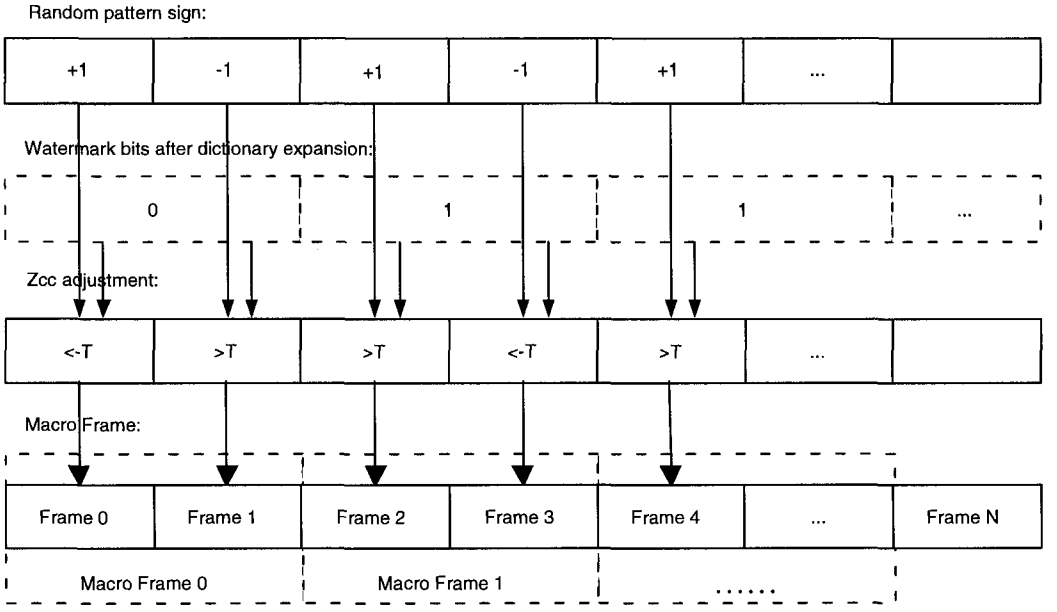

Figure 4.9: Random watermarking pattern sign.

64-bit PN-sequence watermark after dictionary expansion is to be a 64×3=192 bit binary sequence, which will correspondingly embed into 192 macro frames. z_{cc} is the correlation coefficient between the MDCT value in the frame and the random pattern. τ_{cc} is the watermark detection threshold. z_{cc} is adjusted so that its absolute value just exceeds the absolute value of τ_{cc} to make the watermark detectable.

6. Generate Gaussian analysis values for all 32×36 MDCT values in the macro frame. That means, this step generates 1152 Gaussian analysis values.

   The following equations show the detail Gaussian distribution calculation on each macro watermarking frame:

$$f(X_{ij}) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(X_{ij}-\overline{x})^2/(2\sigma^2)} \tag{4.48}$$

$$\overline{x} = \frac{1}{32 \times 36} \sum_{i=1}^{32} \sum_{j=1}^{36} X_{ij} \tag{4.49}$$

$$\sigma = \sqrt{\frac{1}{32 \times 36} \sum_{i=1}^{32} \sum_{j=1}^{36} (X_{ij} - \overline{x})^2} \tag{4.50}$$

where, X_{ij} (i= 1, 2, $\cdots$, 32, j= 1, 2, $\cdots$, 36) represents an MDCT values in a macro watermarking frame.

7. Calculate energy for scalefactor bands in MDCT.

8. Form the final watermark according to random watermark pattern, energy calculation, and Gaussian analysis results. Equ. (4.51) and Equ. (4.52) illustrate the watermark embedding.

$$X'_{ij} = X_{ij} + \Delta X_{ij} \tag{4.51}$$

where, X'_{ij} (i= 1, 2, $\cdots$, 32, j= 1, 2, $\cdots$, 36) represents the watermarked MDCT values in a frame, and

$$\Delta X_{i,j} = \alpha * \frac{log(energy_{ij})}{E} * Sign(X_{ij}) * f(X_{ij}) * C_{ij} \tag{4.52}$$

where, E =100 and

$f(X_{ij})$ is the Gaussian analysis value for X_{ij} in a frame, and

$$C_{ij}= \begin{cases} 0.1, & \text{if } W * Rp_{ij} * Sign(X_{ij}) * X_{ij} > 0 \\ -0.8, & \text{if } W * Rp_{ij} * Sign(X_{ij}) * X_{ij} \leq 0 \end{cases}$$

$$\alpha = \begin{cases} 10^{10\alpha_0 - 2}, & \text{if } \alpha_0 > 0 \\ 0, & \text{if } \alpha_0 \leq 0 \end{cases}$$

$$\alpha_0 = \begin{cases} 2\tau_{cc} - z_{cc}, & \text{if } W * Sign(X_{ij}) > 0 \\ z_{cc} - 2\tau_{cc}, & \text{if } W * Sign(X_{ij}) \leq 0 \end{cases}$$

where, R_p is our watermark random pattern, τ_{cc} is the threshold for detecting watermark, which is used here for embedding strength control.

4.9.3 Watermark Detection

In correlation based watermarking algorithm, correlation coefficient is used to detect the possibility of the watermark existence in a macro frames. We extract watermark after alias detection step and before IMDCT step. It is a blind watermark detection algorithm. The following steps illustrate the watermark extraction in detail.

1. Find the first frame that contains watermark. According to the embedding algorithm, the first bit of watermark is embedded in the first non-zero frame. Therefore, the first non-zero frame should be the first detection candidate.

2. Generate watermark random pattern according to the watermark embedding key.

3. Merge two consecutive frames to one macro frame for watermark detection.

4. Calculate the correlation coefficient using Equ. (4.53) [3]

$$z_{cc}(X, R_p) = \frac{\widetilde{X} \cdot \widetilde{R}_p}{\sqrt{(\widetilde{X} \cdot \widetilde{X})(\widetilde{R}_p \cdot \widetilde{R}_p)}} \tag{4.53}$$

where, X represents MDCT Values in a frame and R_p is our random watermark pattern. $\widetilde{X}$ and $\widetilde{R}_p$ are the mean values of $\overline{X}$ and $\overline{R}_p$, calculated as:

$$\widetilde{X} = (X - \overline{X}) \tag{4.54}$$

$$\widetilde{R}_p = (R_p - \overline{R}_p) \tag{4.55}$$

In Equ. (4.53),

$$\widetilde{X} \cdot \widetilde{R}_p = \sum \widetilde{X}_{ij} \widetilde{R}_{p_{ij}} \tag{4.56}$$

where, i=1, 2, $\cdots$, 32 and j=1, 2, $\cdots$, 18.

In a macro frame, if z_{cc0} is the z_{cc} value for the 1st frame in the macro frame and z_{cc1} is the z_{cc} value for the 2nd frame in the macro frame, we calculate z_{cc}:

$$z_{cc} = z_{cc0} - z_{cc1} \tag{4.57}$$

The correlation coefficient can be interpreted as the inner product of $\widetilde{X}$ and $\widetilde{R}_p$, after each has been normalized to have unit magnitude. This is simply the cosine of the angle between the vectors and is bounded thus:

$-1 \leq z_{cc}(X, Rp) \leq 1$. The detector outputs M_n using the following equation.

$$M_n = \begin{cases} 1, & \text{if } z_{cc}(X, Rp) > \tau_{cc} \\ no \ \ watermark & \text{if } -\tau_{cc} \leq z_{cc}(X, Rp) \leq \tau_{cc} \\ 0, & \text{if } z_{cc}(X, Rp) < -\tau_{cc} \end{cases}$$

where, τ_{cc} is a constant threshold for detecting watermark.

5. Look up the dictionary table to extract final watermark out.

The next chapter will present the experimental results of the two proposed watermarking algorithms with performance evaluation.

Chapter 5

Experimental Results and Evaluations

In this chapter, we conduct experiments to evaluate the proposed watermarking algorithms. The proposed watermarking algorithms are tested under various attacks, such as MP3 compression, low-pass filtering and Gaussian noise addition. The experimental results show that the proposed watermarking algorithms are robust against most common attacks.

5.1 Setup of the Experiments

In our implementation, LAME 3.96.2 open source code is used, which is an MP3 encoder and decoder based on ISO/IEC 11172 standard, Part 3: Audio. The original LAME source code has been modified. We added the proposed watermarking function as a plug-in library in LAME open source code for the implementation and experiments. Therefore, the modified LAME implementation has watermarking capability. The MP3

signals generated by our watermarking function LAME program can be played in a regular MP3 player. The following experiments were conducted on a personal computer with Intel Pentium core 2 duo 2.80 GHZ CPU and 1 GB RAM.

During the experiments, all the original audio signals are in .wav format. Music 1 and Music 2 are classical music (Music 1 and Music 2 are symphony audio with different pitch), Music 3 and Music 4 are pop music (Music 3 is soft pop music and Music 4 is very strong and fast metallic-like music), and Music 5 is a speech signal. These five original audio signals are 45-second long each, and 16-bit signed mono audio signals sampled at 44.1 kHz. A 64-bit watermark sequence is embedded into the original audio signal during MP3 encoding. The watermark is extracted during MP3 decoding process. SNR (Signal-to-Noise Ratio, Equ. (3.3)), PSNR (Peak Signal-to-Noise Ratio, Equ. (3.4)) and BER (Bit Error Rate, Equ. (3.5)) are used as the major indicators to evaluate the performance of the proposed watermarking algorithms. In order to indicate the subjective quality, we use Perceptual Evaluation of Speech Quality (PESQ) to evaluate the quality of watermarked audio signals. We know that PESQ is quality measurement for speech signals. However, we realize that it can also be used to indicate the quality of audio signals. The following tests have been conducted for evaluating the two proposed watermarking algorithms.

5.2 Gaussian Distribution based Watermarking Algorithm

5.2.1 Comparison between the Watermarking Algorithms with and without Adaptive Control

In order to evaluate the benefits of the proposed adaptive watermarking algorithm, we implemented two options in program. The first option is implemented without the adaptive control function, and the other option is implemented with the proposed adaptive control. The same watermark was embedded into five original test signals with the above two options. The PESQ is used to compare the differences between with the adaptive function and without adaptive function.

PESQ is an enhanced perceptual quality measurement for voice quality in telecommunications, especially for end-to-end voice quality testing under real network conditions [20]. The PESQ calculates the MOS to indicate the speech quality. MOS stands for Mean Opinion Score. The PESQ MOS as defined by the ITU recommendation P.862 ranges from 1.0 (worst) up to 4.5 (best) [20].

Table 5.1 and Table 5.2 show the PESQ MOS values for the watermarking algorithm with adaptive control and without adaptive control.

Table 5.1: MOS comparison

	32 kbps			48 kbps			64 kbps		
	adaptive	non-adaptive	BER	adaptive	non-adaptive	BER	adaptive	non-adaptive	BER
Music1	3.775	3.625	0	4.099	3.691	0	4.053	3.939	0
Music2	4.307	4.077	0	4.294	4.105	0	4.356	3.995	0
Music3	4.064	3.294	0	3.854	3.719	0	4.005	3.538	0
Music4	3.938	3.734	0	4.020	3.720	0	3.626	3.617	0
Music5	4.027	3.569	0	3.946	3.557	0	3.948	3.523	0

The experimental results show that, at the same watermark embedding energy, the

Table 5.2: MOS comparison (cont.)

	80 kbps			128 kbps			320 kbps		
	adaptive	non-adaptive	BER	adaptive	non-adaptive	BER	adaptive	non-adaptive	BER
Music1	4.293	3.927	0	4.293	3.951	0	4.287	3.933	0
Music2	4.333	4.000	0	4.351	4.151	0	4.352	4.019	0
Music3	4.162	3.533	0	4.149	3.877	0	4.135	3.522	0
Music4	3.965	3.676	0	4.187	3.524	0	4.162	3.706	0
Music5	3.873	3.565	0	3.970	3.451	0	3.977	3.570	0

algorithm with the adaptive control gives better subjective quality. This is due to the fact we use the human auditory system to adapt the watermark embedding energy to make watermark inaudible.

5.2.2 Different Audio Types

The purpose of this experiment is to demonstrate that the proposed watermarking algorithm works with different types of audio signals.

Watermark is embedded into the five different types of 16-bit signed mono audio signals sampled at 44.1 kHz, each with a length of about 45 seconds in .wav format. Refer to Section 5.1. Table 5.3 gives the experimental results in terms of SNR, PSNR, and BER. The experimental results demonstrate that the proposed watermarking algorithm can work well with different types of audio signals.

Table 5.3: 44.1-kHz 16 bit mono audio

Audio	SNR	PSNR	BER
Music1	32.8202	57.5957	0
Music2	31.0485	56.3534	0
Music3	28.9383	53.3593	0
Music4	29.9110	57.1185	0
Music5	27.8907	55.4103	0

5.2.3 Different Audio Lengths

The purpose of this experiment is to demonstrate that the proposed the Gaussian based watermarking method can work with different length of audio signals. We experimented same classic music (Music1) for Music1-1, Music1-2, Music1-3, Music1-4, and Music1-5 with 20, 30, 45, 60, and 90 seconds respectively.

In the experiment, the same type of audio music (classic) is used and different lengths were chopped by using Cool MP3 Splitter2.2 tool.

To ensure that there are enough MP3 embedding frames available in a piece of audio signal, the minimum length of the audio signal should be greater than 10 seconds. For MP3 audio, whatever the MP3 compression rate is, the same length of audio has the same number of frames. Therefore, for each frame, it costs the same time. For instance, we have a 45-second, 44.1 kHz mono audio. This sample audio is made up by 1753 frames. Each frame consumes 0.026 second (45 seconds/1753). In the proposed watermarking algorithm, watermark is added into $(64\times3)\times2=384$ frames. Totally, it takes $0.026\times384=9.85$ seconds. Therefore, 10 seconds as the minimum length of an audio signal is suggested.

Table 5.4 shows the experimental results in terms of SNR, PSNR, and BER.

Table 5.4: Watermarking on different lengths of audio

Audio	SNR	PSNR	BER	Length
Music1-1	29.6466	54.3978	0	20 sec
Music1-2	30.5416	56.0453	0	30 sec
Music1-3	32.7321	57.5082	0	45 sec
Music1-4	34.4571	60.2020	0	60 sec
Music1-5	35.8003	61.0653	0	90 sec

From the experimental results, we can see that watermark, as a kind of noise added into original audio, impacts audio quality more in shorter audio than in longer audio.

This can be explained by the definitions of SNR and PSNR. Even though the SNR and PSNR show the audio quality are bit difference on different lengths of audio, the experimental results are still acceptable for short audio samples.

5.2.4 MP3 Compression

MP3 compression is one of the most common attacks to audio watermarking. We decompress the watermarked MP3 signal to .wav format, and recompress into MP3 format with different bit rates, 32 kbps, 48 kbps, 64 kbps, 80 kbps, 128 kbps, 160 kbps and 320 kbps. The experimental results are shown in the following tables.

In the following, we will use BER (Bit Error Rate, Equ. (3.5)) to evaluate the robustness of the proposed watermarking algorithm. The value of BER means the ratio between th e number of the error bits and the total number of watermark bits. BER=0 means that there is no difference between the embedded watermark and the extracted watermark, which means no error. According to the experimental results, higher compression ratios may cause bad BER performance, especially when the bite rates are under 32 kbps or 48 kbps. This is because those low bit rates made audio signal lose more information. The lost information may include the embedded watermark, as indicated by the experimental results in terms of BER.

The tables in the following sections will show the audio quality of the watermarked audio signals in terms of SNR, and the robustness of the watermarking algorithm in terms of BER. For all the tables in Section 5.2.4, the columns show the MP3 bit rates used for watermark embedding; while the rows show the MP3 bit rates used for compression attacks. The corresponding SNR and BER values are obtained under the same experimental condition such as embedding and compression bit rates and embedding energy. Usually when SNR values are greater than 20 dB, the audio quality

is acceptable.

5.2.4.1 32-kHz 16-bit mono audio

Table 5.5 shows the BER results on same 32 kHz 16-bit mono audio with different embedding and compression MP3 bit rates. The BER performance on 32-kHz 16-bit mono audio shows that the proposed watermarking algorithm is robust under compression attacks when bit rate is higher than 32 kbps, because 32 kbps compression causes severe information loss.

Table 5.5: BER performance under MP3 compressions on 32-kHz 16-bit mono audio

Compression bit rates	Embedding bit rates						
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	160 kbps	320 kbps
32 kbps	0	33	33	34	36	31	27
48 kbps	0	0	0	0	0	0	0
64 kbps	0	0	0	0	0	0	0
80 kbps	0	0	0	0	0	0	0
128 kbps	0	0	0	0	0	0	0
160 kbps	0	0	0	0	0	0	0
320 kbps	0	0	0	0	0	0	0

Table 5.6 shows the corresponding SNR results on same 32-kHz 16-bit mono audio with different embedding and compression MP3 bit rates. The table shows that the quality of the watermarked audio after further MP3 compressions with different bit rates is still good. This shows that watermarking is not a major factor affecting the quality of watermarked audio.

5.2.4.2 32-kHz 16-bit stereo audio

Table 5.7 shows the BER results on same 32 kHz 16-bit stereo audio with different embedding and compression MP3 bit rates. When test audio is 32-kHz 16-bit stereo, the proposed watermarking algorithm works well with different MP3 compression bit

Table 5.6: SNR performance under MP3 compressions on 32-kHz 16-bit mono audio

Compression bit rates	Embedding bit rates						
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	160 kbps	320 kbps
no compression	22.3373	26.0069	26.1931	28.4042	32.2485	32.9943	33.1434
32 kbps	20.6584	22.2720	22.3965	22.3042	22.3072	22.3615	22.4872
48 kbps	19.7895	21.0235	23.0394	23.3002	23.3276	23.3418	23.3674
64 kbps	20.4331	20.5860	23.1151	24.6970	25.2138	25.2185	25.2870
80 kbps	21.4231	21.7746	23.1794	26.7782	28.3743	28.4943	28.5081
128 kbps	23.1497	23.0908	25.1206	28.0417	32.1333	32.9735	32.9876
160 kbps	23.2878	23.4848	25.2352	28.1990	31.8251	32.8439	32.8690
320 kbps	23.2878	23.4888	25.5789	28.8979	32.0140	32.9796	33.1204

rates except for 32 kbps, 48 kbps and 64 kbps, because these compression rates cause severe information loss to the stereo audio signal.

Table 5.7: BER performance under MP3 compressions on 32-kHz 16-bit stereo audio

Compression bit rates	Embedding bit rates						
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	160 kbps	320 kbps
32 kbps	0	28	32	29	34	31	33
48 kbps	0	0	26	28	31	33	32
64 kbps	0	0	0	25	31	26	29
80 kbps	0	0	0	0	0	0	0
128 kbps	0	0	0	0	0	0	0
160 kbps	0	0	0	0	0	0	0
320 kbps	0	0	0	0	0	0	0

Table 5.8 shows the corresponding SNR results on same 32-kHz 16-bit stereo audio with different embedding and compression MP3 bit rates. The table shows that the quality of the watermarked audio after further MP3 compressions with different bit rates is still good when the embedding bit rate is higher than 32 kbps. This shows that watermarking is not a major factor affecting the quality of watermarked audio.

5.2.4.3 44.1-kHz 16-bit mono audio

Table 5.9 shows the BER results on same 44.1-kHz 16-bit mono audio with different embedding and compression MP3 bit rates. When the test audio is 44.1-kHz 16-bit mono, the proposed watermarking algorithm is robust enough against different MP3

Table 5.8: SNR performance under MP3 compressions on 32-kHz 16-bit stereo audio

Compression bit rates	Embedding bit rates						
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	160 kbps	320 kbps
no compression	23.1443	23.3461	28.9050	29.3707	29.7727	31.2730	32.8864
32 kbps	18.4394	21.0976	22.1108	22.8733	23.1592	23.1650	23.3620
48 kbps	18.9344	20.6509	22.6591	24.5683	26.0702	26.1788	26.3707
64 kbps	20.9404	22.6480	22.9851	24.7061	28.4403	28.6837	28.8826
80 kbps	18.8770	20.8387	21.5013	22.8619	27.8733	28.4205	29.3707
128 kbps	18.9172	20.8561	20.9575	21.9171	23.5580	28.8892	29.7629
160 kbps	18.6163	20.4831	20.9627	21.7955	22.8463	27.1135	28.7967
320 kbps	18.9287	21.0382	21.5735	22.7251	24.4382	28.4612	32.7124

compression bit rates except for 32 kbps and 48 kbps, because these compression rates cause severe information loss.

Table 5.9: BER performance under MP3 compressions on 44.1-kHz 16-bit mono audio

Compression bit rates	Embedding bit rates						
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	160 kbps	320 kbps
32 kbps	0	30	28	29	28	32	31
48 kbps	0	0	31	32	26	31	27
64 kbps	0	0	0	0	0	0	0
80 kbps	0	0	0	0	0	0	0
128 kbps	0	0	0	0	0	0	0
160 kbps	0	0	0	0	0	0	0
320 kbps	0	0	0	0	0	0	0

Table 5.10 shows the corresponding SNR results on same 44.1-kHz 16-bit mono audio. The SNR results on 44.1-kHz 16-bit mono audio show that the audio quality after watermarking is good.

5.2.4.4 44.1-kHz 16-bit stereo audio

Table 5.11 shows the BER results on same 44.1-kHz 16-bit stereo audio with different embedding and compression MP3 bit rates. For 44.1-kHz 16-bit stereo audio, the proposed watermarking algorithm works well with different compression rates except for 32 kbps, 48 kbps, 64 kbps and 80 kbps, because these compression rates cause severe

Table 5.10: SNR performance under MP3 compressions on 44.1-kHz 16-bit mono audio

Compression bit rates	Embedding bit rates						
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	160 kbps	320 kbps
no compression	25.7346	26.6374	27.7858	30.1687	35.2809	35.9870	36.3191
32 kbps	20.7549	22.4065	22.5603	22.6245	24.6715	24.7060	24.8282
48 kbps	20.5787	22.5029	24.1013	24.5603	24.6283	24.6418	24.8279
64 kbps	23.0785	23.7858	25.2505	26.8202	27.8771	27.9281	27.9675
80 kbps	23.9337	24.3692	25.8738	27.1825	30.0275	30.1223	30.1828
128 kbps	25.4886	25.8442	27.0555	27.2825	34.0529	35.6134	36.3066
160 kbps	25.6880	25.7603	27.2736	29.2441	34.8192	35.6147	36.0868
320 kbps	25.6058	25.8868	27.2155	28.9128	34.9026	35.8542	36.3075

information loss to the audio.

Table 5.11: BER performance under MP3 compressions on 44.1-kHz 16-bit stereo audio

Compression bit rates	Embedding bit rates						
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	160 kbps	320 kbps
32 kbps	0	33	29	34	33	34	32
48 kbps	0	0	26	30	31	30	33
64 kbps	0	0	0	0	33	28	32
80 kbps	0	0	0	0	28	28	29
128 kbps	0	0	0	0	0	0	0
160 kbps	0	0	0	0	0	0	0
320 kbps	0	0	0	0	0	0	0

Table 5.12 shows the corresponding SNR results on same 44.1-kHz 16-bit stereo audio. The SNR results on 44.1-kHz 16-bit stereo audio show that the audio quality after watermarking is generally good.

5.2.4.5 48-kHz 16-bit mono audio

Table 5.13 shows the BER results on same 48-kHz 16-bit mono audio with different embedding and compression MP3 bit rates. For 48-kHz 16-bit mono audio, the proposed watermarking method works well with different compression rates except for 32 kbps and 48 kbps, because these compression rates cause severe information loss to the host audio and watermark.

Table 5.12: SNR performance under MP3 compressions on 44.1-kHz 16-bit stereo audio

Compression bit rates	Embedding bit rates						
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	160 kbps	320 kbps
no compression	22.6532	25.5589	28.4828	29.6021	33.9229	34.1288	35.8941
32 kbps	20.8294	20.9723	21.8047	22.5972	22.5977	24.7263	24.9282
48 kbps	18.6568	20.8614	22.5978	24.1296	25.4122	25.5118	26.2106
64 kbps	20.7381	22.7718	23.4340	24.8524	28.0121	28.3820	29.7828
80 kbps	18.9874	21.2577	22.2107	24.5093	24.5232	28.7099	29.5752
128 kbps	20.6944	23.2164	24.0904	26.3527	29.3018	30.5191	33.7740
160 kbps	20.8021	23.3485	24.3626	26.7889	28.7159	29.4824	34.9151
320 kbps	20.7318	23.3423	24.8427	27.0608	29.3622	30.1894	35.5488

Table 5.13: BER performance under MP3 compressions on 48-kHz 16-bit mono audio

Compression bit rates	Embedding bit rates						
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	160 kbps	320 kbps
32 kbps	0	31	31	28	31	34	32
48 kbps	0	0	24	24	34	31	28
64 kbps	0	0	0	0	0	0	0
80 kbps	0	0	0	0	0	0	0
128 kbps	0	0	0	0	0	0	0
160 kbps	0	0	0	0	0	0	0
320 kbps	0	0	0	0	0	0	0

Table 5.14 shows the corresponding SNR results on same 48-kHz 16-bit mono audio with different embedding and compression bit rates. The SNR results on 48-kHz 16-bit mono audio show that the audio quality after watermarking is good.

5.2.4.6 48-kHz 16-bit stereo audio

Table 5.15 shows the BER results on same 48-kHz 16-bit stereo audio with different embedding and compression bit rates. For 48-kHz 16-bit stereo audio, the proposed watermarking algorithm works well when compression rates are equal or higher than 120 kbps.

Table 5.16 shows the corresponding SNR results on same 48-kHz 16-bit stereo audio. The audio quality after watermarking and compression attacks with different bit rates

Table 5.14: SNR performance under MP3 compressions on 48-kHz 16 bit mono audio

Compression bit rates	Embedding bit rates						
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	160 kbps	320 kbps
no compression	25.9932	26.5442	26.6833	28.8700	34.4373	34.8714	35.6575
32 kbps	20.5809	22.3514	22.4201	22.4376	22.4218	22.4459	24.2903
48 kbps	19.7405	20.9563	22.8457	23.3329	23.2908	23.2922	23.2946
64 kbps	22.3883	22.8352	23.8882	25.9596	26.6129	26.6404	26.7388
80 kbps	23.2694	23.4878	24.0155	25.9001	28.7246	28.7717	28.7788
128 kbps	25.1570	25.1580	25.7133	27.3481	33.2418	34.1114	34.4536
160 kbps	25.2140	25.4380	26.2317	28.2402	33.1980	34.2095	34.9024
320 kbps	25.8321	25.5730	26.2678	28.6590	34.3008	34.8331	35.2315

Table 5.15: BER performance under MP3 compressions on 48-kHz 16-bit stereo audio

Compression bit rates	Embedding bit rates						
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	160 kbps	320 kbps
32 kbps	0	28	33	34	33	32	30
48 kbps	0	0	33	31	31	30	29
64 kbps	0	0	0	0	33	32	28
80 kbps	0	0	0	0	28	28	34
128 kbps	0	0	0	0	0	0	0
160 kbps	0	0	0	0	0	0	0
320 kbps	0	0	0	0	0	0	0

is generally good.

5.2.5 Low-Pass Filtering

CoolEditPro v2.1 tool is used to test the robustness of the proposed algorithm against low-pass filtering. According to Human Auditory System, the audible range is from 20 Hz to 20 kHz, therefore, the cutoff frequency was set to 20 kHz, and 10 kHz in this experiment. Tables 5.17 and 5.18 show the performance against low-pass filtering attacks.

In Table 5.17, the cut-off frequency is 20 kHz. The BERs (Bit Error Rates) for Music 4 and Music 5 are not 0 %. Music 4 is a very fast and strong pop music. Music 5 is a speech audio, whose energy is concentrated in the middle frequencies, and it does

Table 5.16: SNR performance under MP3 compressions on 48-kHz 16 bit stereo audio

Compression bit rates	Embedding bit rates						
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	160 kbps	320 kbps
no compression	23.0424	25.8910	28.7100	28.7863	33.0051	33.8748	35.3328
32 kbps	18.3188	21.0773	22.1422	22.9118	23.0230	23.0280	23.8826
48 kbps	18.7226	20.5497	22.4985	24.4320	25.8462	25.8925	26.3008
64 kbps	20.6323	22.4866	22.2880	24.4402	28.1481	28.4718	29.9800
80 kbps	18.7668	20.6664	21.1253	22.9790	27.4625	27.9562	28.8344
128 kbps	21.2228	23.2213	23.5701	24.8480	27.3531	29.8182	32.9565
160 kbps	21.1774	23.5465	25.5288	25.5428	27.5713	29.3256	34.9087
320 kbps	21.0212	23.2139	24.1182	25.6842	27.2521	29.3520	34.6650

Table 5.17: Performance against low-pass filtering (cut-off frequency = 20 kHz)

Audio	SNR	PSNR	BER
Music1	26.6670	50.0690	0
Music2	20.3884	47.9038	0
Music3	22.4124	48.3351	0
Music4	15.9261	37.3472	5
Music5	16.3452	39.0844	1

not show much rhythm.

5.2.6 Gaussian Noise

CoolEditPro v2.1 tool is used to generate 10 dB and 20 dB Gaussian noise. Then we add this period of noise on the watermarked audio in order to simulate the transaction through a communication channel with noise and also to simulate the attack with a noise added to watermarked audio intending to blur the watermark. Table 5.19 shows when the Gaussian noise is 10 dB, the BER value is low, indicating a good performance.

Table 5.18: Performance against low-pass filtering (cut-off frequency = 10 kHz)

Audio	SNR	PSNR	BER
Music1	18.1589	43.6661	30
Music2	13.1754	35.7874	33
Music3	23.4796	46.8824	30
Music4	12.4801	40.5195	31
Music5	10.5817	35.1006	30

Table 5.20 shows when the Gaussian noise is 20 dB, the BER value is high, indicating unsatisfactory performance.

Table 5.19: Performance against Gaussian noise (10 dB)

Audio	BER
Music1	0
Music2	0
Music3	0
Music4	4
Music5	1

Table 5.20: Performance against Gaussian noise (20 dB)

Audio	BER
Music1	28
Music2	30
Music3	30
Music4	31
Music5	33

5.2.7 Different Compression Rates

The purpose of this experiment is to ensure that the proposed watermarking algorithm can work on audio signals with different compression rates. The most common used compression rates are 32 kbps, 48 kbps, 64 kbps, 80 kbps, 128 kbps, and 320 kbps.

We use WINAMP tool to obtain audio signals in six most common formats with a length of 45 seconds and with different MP3 rates: 32k-16bit-mono.wav, 32k-16bit-stereo.wav, 44.1k-16bit-mono.wav, 44.1k-16bit-stereo.wav, 48k-16bit-mono.wav, 48k-16bit-stereo.wav. The same original audio compression rate was used in both watermark embedding and detection. The following tables show our test results: the proposed watermarking algorithm had good performance in most common audio formats with their own compression rates.

Table 5.21: An audio with various compression rate coverage tests (32 kbps)

Original Wav file	Transforming to MP3 format	BER	SNR	PSNR
32kHz, 16bit, Mono, 32kbps	32kbps	0	22.3373	49.8485
32kHz, 16bit, Stereo, 32kbps	32kbps	0	23.1443	50.7015
44.1kHz, 16bit, Mono, 32kbps	32kbps	0	22.6671	50.1726
44.1kHz, 16bit, Stereo, 32kbps	32kbps	0	22.6532	50.2224
48kHz, 16bit, Mono, 32kbps	32kbps	0	22.3091	49.8131
48kHz, 16bit, Stereo, 32kbps	32kbps	0	23.0424	50.6241

Table 5.22: An audio with various compression rate coverage tests (48 kbps)

Original Wav file	Transforming to MP3 format	BER	SNR	PSNR
32kHz, 16bit, Mono, 48kbps	48kbps	0	26.0069	53.6171
32kHz, 16bit, Stereo, 48kbps	48kbps	0	23.3461	50.8605
44.1kHz, 16bit, Mono, 48kbps	48kbps	0	24.6413	52.1522
44.1kHz, 16bit, Stereo, 48kbps	48kbps	0	25.5589	53.1503
48kHz, 16bit, Mono, 48kbps	48kbps	0	23.2530	50.7610
48kHz, 16bit, Stereo, 48kbps	48kbps	0	25.8910	53.4895

Figure 5.1 summarizes the results of this experimental test. In Figure 5.1, the x-axis represents six most common audio formats such as 32k-16bit-mono.wav, 32k-16bit-stereo.wav, 44.1k-16bit-mono.wav, 44.1k-16bit-stereo.wav, 48k-16bit-mono.wav and 48k-16bit-stereo.wav with different MP3 bit rates such as 32 kbps, 48 kbps, 64 kbps, 80 kbps, 128 kbps and 320 kbps, totally 36 (6×6) combinations. The y-axis represents the SNR and PSNR values. All the BER values are 0.

The above experimental results shows that the proposed Gaussian distribution based watermarking algorithm's performance is stable under most common MP3 compression bit rates.

Table 5.23: An audio with various compression rate coverage tests (64 kbps)

Original Wav file	Transforming to MP3 format	BER	SNR	PSNR
32kHz, 16bit, Mono, 64kbps	64kbps	0	25.1943	52.7093
32kHz, 16bit, Stereo, 64kbps	64kbps	0	28.9050	56.6412
44.1kHz, 16bit, Mono, 64kbps	64kbps	0	27.7858	55.3049
44.1kHz, 16bit, Stereo, 64kbps	64kbps	0	28.4824	56.2461
48kHz, 16bit, Mono, 64kbps	64kbps	0	26.6833	54.1968
48kHz, 16bit, Stereo, 64kbps	64kbps	0	28.7100	56.4380

Table 5.24: An audio with various compression rate coverage tests (80 kbps)

Original Wav file	Transforming to MP3 format	BER	SNR	PSNR
32kHz, 16bit, Mono, 80kbps	80kbps	0	28.4042	55.9255
32kHz, 16bit, Stereo, 80kbps	80kbps	0	29.3707	57.2880
44.1kHz, 16bit, Mono, 80kbps	80kbps	0	30.1687	57.6896
44.1kHz, 16bit, Stereo, 80kbps	80kbps	0	29.6021	57.6651
48kHz, 16bit, Mono, 80kbps	80kbps	0	28.8700	56.3869
48kHz, 16bit, Stereo, 80kbps	80kbps	0	28.7863	56.7461

Table 5.25: An audio with various compression rate coverage tests (128 kbps)

Original Wav file	Transforming to MP3 format	BER	SNR	PSNR
32kHz, 16bit, Mono, 128kbps	128kbps	0	32.2485	59.7722
32kHz, 16bit, Stereo, 128kbps	128kbps	0	29.7727	58.5106
44.1kHz, 16bit, Mono, 128kbps	128kbps	0	35.2809	62.8047
44.1kHz, 16bit, Stereo, 128bps	128kbps	0	33.9229	62.6932
48kHz, 16bit, Mono, 128kbps	128kbps	0	34.4373	61.9587
48kHz, 16bit, Stereo, 128kbps	128kbps	0	33.0051	61.5529

5.3 Correlation based Watermarking Algorithm

Table 5.27 and Table 5.28 show the PESQ MOS values for the watermarking method 2 with adaptive control and without adaptive control.

The experimental results show that the algorithm with the adaptive control gives better subjective quality.

Table 5.26: An audio with various compression rate coverage tests (320 kbps)

Original Wav file	Transforming to MP3 format	BER	SNR	PSNR
32kHz, 16bit, Mono, 320kbps	320kbps	0	33.1434	60.6674
32kHz, 16bit, Stereo, 320kbps	320kbps	0	32.6806	63.1983
44.1kHz, 16bit, Mono, 320kbps	320kbps	0	36.3191	63.8432
44.1kHz, 16bit, Stereo, 320kbps	320kbps	0	35.8941	66.4172
48kHz, 16bit, Mono, 320kbps	320kbps	0	35.2125	62.7343
48kHz, 16bit, Stereo, 320kbps	320kbps	0	34.7248	65.2052

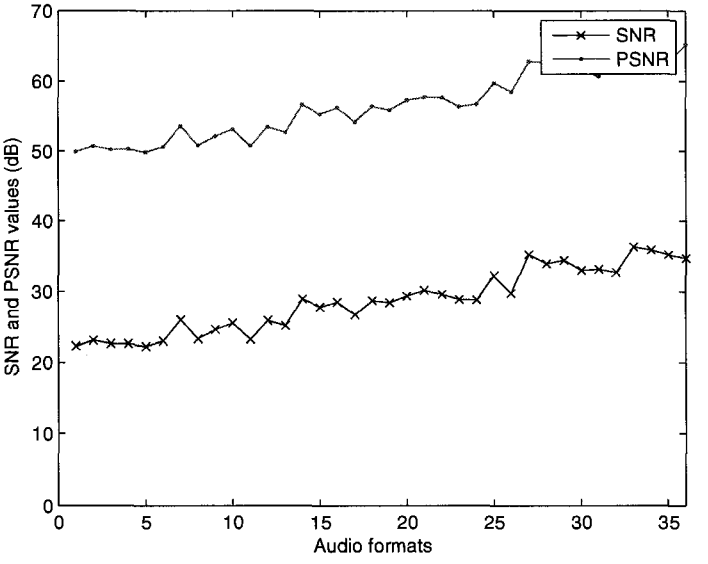

Figure 5.1: Summary of an audio with various compression rate coverage tests

Table 5.27: MOS comparison

	32 kbps			48 kbps			64 kbps		
	adaptive	non-adaptive	BER	adaptive	non-adaptive	BER	adaptive	non-adaptive	BER
Music1	3.741	3.697	0	3.557	3.418	0	3.548	3.334	0
Music2	3.642	3.606	0	3.430	3.465	0	3.499	3.288	0
Music3	4.033	4.032	0	3.954	4.106	0	4.055	3.998	0
Music4	3.581	3.498	0	3.344	3.228	0	3.447	3.355	0
Music5	3.726	3.617	0	3.735	3.713	0	3.907	3.852	0

5.3.1 Different Audio Types

Watermark is embedded into the five different types of 16-bit signed mono audio signals sampled at 44.1 kHz, which is a standard CD audio quality format, each with a length of about 45 seconds in .wav format. Table 5.29 gives the experimental results in terms of SNR, PSNR, and BER. The experimental results demonstrate that the proposed watermarking algorithm 2 can work well with different types of audio signals.

Table 5.28: MOS comparison (cont.)

	80 kbps			128 kbps			320 kbps		
	adaptive	non-adaptive	BER	adaptive	non-adaptive	BER	adaptive	non-adaptive	BER
Music1	3.545	3.397	0	3.534	3.497	0	3.561	3.443	0
Music2	3.516	3.349	0	3.456	3.367	0	3.413	3.355	0
Music3	4.083	4.028	0	4.165	3.997	0	4.160	3.956	0
Music4	3.498	3.344	0	3.449	3.445	0	3.411	3.399	0
Music5	3.864	3.837	0	3.964	3.809	0	3.975	3.833	0

Table 5.29: 44.1-kHz 16 bit mono audio

Audio	SNR	PSNR	BER
Music1	27.2745	52.0416	0
Music2	22.1492	49.3368	0
Music3	24.6289	48.0308	0
Music4	24.6412	46.0605	0
Music5	23.2992	49.2118	0

5.3.2 Different Audio Lengths

The purpose of this experiment is to demonstrate that the proposed the correlation based watermarking method can work with different length of audio signals. We experimented same classic music (Music1) with a length of 20, 30, 45, 60, and 90 seconds respectively, as shown as Music1-1, Music1-2, Music1-3, Music1-4, and Music1-5 in Table 5.30. Table 5.30 illustrated the experimental results in terms of SNR, PSNR, and BER.

Table 5.30: Watermarking on different lengths of audios

Audio	SNR	PSNR	BER	Length
Music1-1	21.2128	48.5937	0	20 sec
Music1-2	23.9362	50.0139	0	30 sec
Music1-3	23.4565	49.3692	0	45 sec
Music1-4	25.5367	52.9072	0	60 sec
Music1-5	29.0225	55.3689	0	90 sec

From the experimental results, we can see that watermark, as a kind of noise added into original audio, impacts audio quality more in shorter audio than in longer audio.

This can be explained by the definitions of SNR and PSNR. Even though the SNR and PSNR show the audio quality are bit difference on different lengths of audio, the experimental results are still acceptable for short audio samples.

5.3.3 MP3 Compression

MP3 compression is one of the most common attacks to audio watermarking. We tested our algorithm on 16-bit mono and stereo audio signals sampled at 32-kHz, 44.1-kHz, 48-kHz, 64-kbps, 80-kbps, 128-kbps and 320-kbps.

5.3.3.1 32-kHz 16-bit mono audio

Table 5.31 shows the BER results on same 32 kHz 16-bit mono audio with different embedding and compression MP3 bit rates. The BER performance on 32-kHz 16-bit mono audio shows that the proposed watermarking algorithm is robust under compression attacks when bit rate is higher than 32 kbps, because 32 kbps compression causes severe information loss.

Table 5.31: BER performance under MP3 compressions on 32-kHz 16-bit mono audio

Embedding bit rates	Compression bit rates					
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	320 kbps
32 kbps	Fail	0	0	0	0	0
48 kbps	Fail	Fail	Fail	0	0	0
64 kbps	Fail	Fail	Fail	0	0	0
80 kbps	Fail	Fail	Fail	0	0	0
128 kbps	Fail	Fail	Fail	0	0	0
320 kbps	Fail	Fail	Fail	0	0	0

Table 5.32 shows the corresponding SNR results on same 32 kHz 16-bit mono audio with different embedding and compression MP3 bit rates. The table shows that the quality of the watermarked audio after further MP3 compressions with different bit

rates is still good. This shows that watermarking is not a major factor affecting the quality of watermarked audio.

Table 5.32: SNR performance under MP3 compressions on 32-kHz 16-bit mono audio

Embedding bit rates	Compression bit rates					
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	320 kbps
no compression	18.8653	21.7776	22.7366	24.3428	25.1035	28.9023
32 kbps	Fail	21.4854	21.8621	22.0252	21.4590	21.7789
48 kbps	Fail	Fail	Fail	22.0487	21.5622	21.8823
64 kbps	Fail	Fail	Fail	22.1600	24.2708	23.3634
80 kbps	Fail	Fail	Fail	22.3378	24.6953	26.5434
128 kbps	Fail	Fail	Fail	23.6756	24.6319	26.6472
320 kbps	Fail	Fail	Fail	23.3570	24.4419	28.4825

5.3.3.2 32-kHz 16-bit stereo audio

Table 5.33 shows the BER results on same 32 kHz 16-bit stereo audio with different embedding and compression MP3 bit rates. When test audio is 32-kHz 16-bit stereo, the proposed watermarking algorithm works well with different MP3 compression bit rates except for 32 kbps, 48 kbps and 64 kbps, because these compression rates cause severe information loss to the stereo audio signal.

Table 5.33: BER performance under MP3 compressions on 32-kHz 16-bit stereo audio

Embedding bit rates	Compression bit rates					
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	320 kbps
32 kbps	Fail	0	0	0	0	0
48 kbps	Fail	Fail	Fail	0	0	0
64 kbps	Fail	Fail	Fail	0	0	0
80 kbps	Fail	Fail	Fail	0	0	0
128 kbps	Fail	Fail	Fail	0	0	0
320 kbps	Fail	Fail	Fail	0	0	0

Table 5.34 shows the corresponding SNR results on same 32-kHz 16-bit stereo audio with different embedding and compression MP3 bit rates. The table shows that the

quality of the watermarked audio after further MP3 compressions with different bit rates is still good when the embedding bit rate is higher than 32 kbps. This shows that watermarking is not a major factor affecting the quality of watermarked audio.

Table 5.34: SNR performance under MP3 compressions on 32-kHz 16-bit stereo audio

Embedding bit rates	Compression bit rates					
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	320 kbps
no compression	14.2355	16.7876	18.1284	23.6686	25.5690	28.4433
32 kbps	Fail	16.7248	17.9643	18.0982	18.0995	18.4562
48 kbps	Fail	Fail	Fail	21.6743	22.3218	22.1725
64 kbps	Fail	Fail	Fail	23.3478	25.3650	25.6549
80 kbps	Fail	Fail	Fail	23.4619	23.7459	24.7981
128 kbps	Fail	Fail	Fail	22.4521	24.2057	25.2846
320 kbps	Fail	Fail	Fail	20.1115	24.4648	27.4994

5.3.3.3 44.1-kHz 16-bit mono audio

Table 5.35 shows the BER results on same 44.1-kHz 16-bit mono audio with different embedding and compression MP3 bit rates. When the test audio is 44.1-kHz 16-bit mono, the proposed watermarking algorithm is robust enough against different MP3 compression bit rates except for 32 kbps and 48 kbps, because these compression rates cause severe information loss.

Table 5.35: BER performance under MP3 compressions on 44.1-kHz 16-bit mono audio

Embedding bit rates	Compression bit rates					
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	320 kbps
32 kbps	Fail	0	0	0	0	0
48 kbps	Fail	Fail	Fail	0	0	0
64 kbps	Fail	Fail	Fail	0	0	0
80 kbps	Fail	Fail	Fail	0	0	0
128 kbps	Fail	Fail	Fail	0	0	0
320 kbps	Fail	Fail	Fail	0	0	0

Table 5.36 shows the corresponding SNR results on same 44.1-kHz 16-bit mono audio. The SNR results on 44.1-kHz 16-bit mono audio show that the audio quality

after watermarking is good.

Table 5.36: SNR performance under MP3 compressions on 44.1-kHz 16-bit mono audio

Embedding bit rates	Compression bit rates					
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	320 kbps
no compression	18.9509	20.3344	24.2931	23.8843	24.9346	28.8979
32 kbps	Fail	20.1716	22.5829	22.6715	24.3072	24.4872
48 kbps	Fail	Fail	Fail	22.0312	22.1154	23.0318
64 kbps	Fail	Fail	Fail	22.1600	23.0702	23.1670
80 kbps	Fail	Fail	Fail	22.4849	23.1135	24.8826
128 kbps	Fail	Fail	Fail	22.8561	24.8619	26.1728
320 kbps	Fail	Fail	Fail	23.5735	24.8883	28.7629

5.3.3.4 44.1-kHz 16-bit stereo audio

Table 5.37 shows the BER results on same 44.1-kHz 16-bit stereo audio with different embedding and compression MP3 bit rates. For 44.1-kHz 16-bit stereo audio, the proposed watermarking algorithm works well with different compression rates except for 32 kbps, 48 kbps, 64 kbps and 80 kbps, because these compression rates cause severe information loss to the audio.

Table 5.37: BER performance under MP3 compressions on 44.1-kHz 16-bit stereo audio

Embedding bit rates	Compression bit rates					
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	320 kbps
32 kbps	Fail	0	0	0	0	0
48 kbps	Fail	Fail	Fail	0	0	0
64 kbps	Fail	Fail	Fail	0	0	0
80 kbps	Fail	Fail	Fail	0	0	0
128 kbps	Fail	Fail	Fail	0	0	0
320 kbps	Fail	Fail	Fail	0	0	0

Table 5.38 shows the corresponding SNR results on same 44.1-kHz 16-bit stereo audio. The SNR results on 44.1-kHz 16-bit stereo audio show that the audio quality after watermarking is generally good.

Table 5.38: SNR performance under MP3 compressions on 44.1-kHz 16-bit stereo audio

Embedding bit rates	Compression bit rates					
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	320 kbps
no compression	14.6482	19.9972	20.5554	25.8890	27.8844	29.9288
32 kbps	Fail	19.9723	20.1987	20.1108	22.0382	22.1683
48 kbps	Fail	Fail	Fail	22.3959	22.6481	22.9103
64 kbps	Fail	Fail	Fail	23.8578	24.4849	25.5440
80 kbps	Fail	Fail	Fail	23.2512	25.5160	28.6051
128 kbps	Fail	Fail	Fail	24.6160	26.7746	28.7058
320 kbps	Fail	Fail	Fail	25.7362	27.8427	29.8426

5.3.3.5 48-kHz 16-bit mono audio

Table 5.39 shows the BER results on same 48-kHz 16-bit mono audio with different embedding and compression MP3 bit rates. For 48-kHz 16-bit mono audio, the proposed watermarking method works well with different compression rates except for 32 kbps and 48 kbps, because these compression rates cause severe information lost to the host audio and watermark.

Table 5.39: BER performance under MP3 compressions on 48-kHz 16-bit mono audio

Embedding bit rates	Compression bit rates					
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	320 kbps
32 kbps	Fail	0	0	0	0	0
48 kbps	Fail	Fail	Fail	0	0	0
64 kbps	Fail	Fail	Fail	0	0	0
80 kbps	Fail	Fail	Fail	0	0	0
128 kbps	Fail	Fail	Fail	0	0	0
320 kbps	Fail	Fail	Fail	0	0	0

Table 5.40 shows the corresponding SNR results on same 48-kHz 16-bit mono audio with different embedding and compression bit rates. The SNR results on 48-kHz 16-bit mono audio show that the audio quality after watermarking is good.

Table 5.40: SNR performance under MP3 compressions on 48-kHz 16 bit mono audio

Embedding bit rates	Compression bit rates					
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	320 kbps
no compression	18.2846	22.2376	23.7264	26.9718	31.3724	33.5495
32 kbps	Fail	21.4934	22.3655	22.9510	22.2778	22.8397
48 kbps	Fail	Fail	Fail	21.7357	22.8614	23.6188
64 kbps	Fail	Fail	Fail	22.8362	24.0289	28.1336
80 kbps	Fail	Fail	Fail	22.5909	24.8733	29.7836
128 kbps	Fail	Fail	Fail	23.3167	25.5880	29.8765
320 kbps	Fail	Fail	Fail	26.2988	31.3487	33.4501

5.3.3.6 48-kHz 16-bit stereo audio

Table 5.41 shows the BER results on same 48-kHz 16-bit stereo audio with different embedding and compression bit rates. For 48-kHz 16-bit stereo audio, the proposed watermarking algorithm works well when compression rates are equal or higher than 120 kbps.

Table 5.41: BER performance under MP3 compressions on 48-kHz 16-bit stereo audio

Embedding bit rates	Compression bit rates					
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	320 kbps
32 kbps	Fail	0	0	0	0	0
48 kbps	Fail	Fail	Fail	0	0	0
64 kbps	Fail	Fail	Fail	0	0	0
80 kbps	Fail	Fail	Fail	0	0	0
128 kbps	Fail	Fail	Fail	0	0	0
320 kbps	Fail	Fail	Fail	0	0	0

Table 5.42 shows the corresponding SNR results on same 48-kHz 16-bit stereo audio. The audio quality after watermarking and compression attacks with different bit rates is generally good.

5.3.4 Low-Pass Filtering

Table 5.43 showed the performance against the low-pass filtering attack:

Table 5.42: SNR performance under MP3 compressions on 48-kHz 16 bit stereo audio

Embedding bit rates	Compression bit rates					
	32 kbps	48 kbps	64 kbps	80 kbps	128 kbps	320 kbps
no compression	18.3437	20.5152	24.7375	28.3856	32.8972	36.0388
32 kbps	Fail	20.5084	21.3377	21.8802	22.0913	22.9828
48 kbps	Fail	Fail	Fail	21.2598	23.1317	24.8864
64 kbps	Fail	Fail	Fail	22.2346	24.2882	29.7818
80 kbps	Fail	Fail	Fail	22.6137	25.9460	28.6766
128 kbps	Fail	Fail	Fail	23.1207	27.2813	32.2315
320 kbps	Fail	Fail	Fail	27.1230	32.8324	35.7178

Table 5.43: Against low-pass filtering (20 kHz)

Audio through low-pass filtering	SNR	PSNR	BER
Music1	22.3325	45.7217	0
Music2	21.1991	45.0388	0
Music3	20.7889	44.0128	0
Music4	11.2011	39.2220	6
Music5	17.6766	43.1645	1

The BERs (Bit Error Rates) for Music 4 and 5 are not 0 %. Music 4 is a very fast and strong pop music. Music 5 is a speech audio, whose energy is concentrated in the middle frequencies, and it does not show much rhythm.

5.3.5 Gaussian Noise

We set Gaussian noise to be 10 dB Table 5.44 gives the results:

Table 5.44: Against Gaussian noise (10 dB)

Audio through Gaussian noise	BER
Music1	0
Music2	0
Music3	0
Music4	5
Music5	1

The same reason as that in low-pass filter, the BERs (Bit Error Rates) for Music 4

and 5 are not 0 %. Music 4 is a very fast and strong pop music. Music 5 is a speech audio, whose energy is concentrated in the middle frequencies, and it does not show much rhythm.

5.3.6 Different Compression Rates

To demonstrate that the proposed algorithm works with various MP3 compression rates, an audio signal of 45 seconds with different features is used to form 6 different test audios: 32k-16bit-mono.wav, 32k-16bit-stereo.wav, 44.1k-16bit-mono.wav, 44.1k-16bit-stereo.wav, 48k-16bit-mono.wav, 48k-16bit-stereo.wav. The same original audio compression rate was used in both watermark embedding and detection. The following tables show our test results. The proposed watermarking algorithm has good performance in most common audio formats.

Table 5.45: An audio with various compression rate coverage tests (32 kbps)

Original Wav file	Transforming to MP3 format	BER	SNR	PSNR
32kHz, 16bit, Mono, 32kbps	32kbps	0	18.8653	44.7358
32kHz, 16bit, Stereo, 32kbps	32kbps	0	14.2355	41.5622
44.1kHz, 16bit, Mono, 32kbps	32kbps	0	18.9509	44.8238
44.1kHz, 16bit, Stereo, 32kbps	32kbps	0	14.6482	41.9146
48kHz, 16bit, Mono, 32kbps	32kbps	0	18.2846	44.1399
48kHz, 16bit, Stereo, 32kbps	32kbps	0	13.5154	40.9588

Table 5.46: An audio with various compression rate coverage tests (48 kbps)

Original Wav file	Transforming to MP3 format	BER	SNR	PSNR
32kHz, 16bit, Mono, 48kbps	48kbps	0	20.3542	46.2454
32kHz, 16bit, Stereo, 48kbps	48kbps	0	16.3511	44.1504
44.1kHz, 16bit, Mono, 48kbps	48kbps	0	20.1200	46.0100
44.1kHz, 16bit, Stereo, 48kbps	48kbps	0	15.7772	43.7086
48kHz, 16bit, Mono, 48kbps	48kbps	0	19.2735	45.1579
48kHz, 16bit, Stereo, 48kbps	48kbps	0	17.3437	45.1874

Figure 5.2 summarizes the results for the correlation based watermarking algorithm. In Figure 5.2, the x-axis represents six most common audio formats (32k-

Table 5.47: An audio with various compression rate coverage tests (64 kbps)

Original Wav file	Transforming to MP3 format	BER	SNR	PSNR
32kHz, 16bit, Mono, 64kbps	64kbps	0	19.7366	45.6279
32kHz, 16bit, Stereo, 64kbps	64kbps	0	17.2848	45.4848
44.1kHz, 16bit, Mono, 64kbps	64kbps	0	24.2931	50.2068
44.1kHz, 16bit, Stereo, 64kbps	64kbps	0	18.2757	46.5110
48kHz, 16bit, Mono, 64kbps	64kbps	0	23.7264	49.6356
48kHz, 16bit, Stereo, 64kbps	64kbps	0	17.3343	45.5880

Table 5.48: An audio with various compression rate coverage tests (80 kbps)

Original Wav file	Transforming to MP3 format	BER	SNR	PSNR
32kHz, 16bit, Mono, 80kbps	80kbps	0	18.0328	43.9012
32kHz, 16bit, Stereo, 80kbps	80kbps	0	20.1634	48.7065
44.1kHz, 16bit, Mono, 80kbps	80kbps	0	23.0027	48.9126
44.1kHz, 16bit, Stereo, 80kbps	80kbps	0	20.1845	48.7743
48kHz, 16bit, Mono, 80kbps	80kbps	0	23.3064	49.2118
48kHz, 16bit, Stereo, 80kbps	80kbps	0	19.7054	48.2309

16bit-mono.wav, 32k-16bit-stereo.wav, 44.1k-16bit-mono.wav, 44.1k-16bit-stereo.wav, 48k-16bit-mono.wav and 48k-16bit-stereo.wav) with different MP3 bit rates such as 32 kbps, 48 kbps, 64 kbps, 80 kbps, 128 kbps and 320 kbps, totally 36 (6×6) combinations. The y-axis represents the SNR and PSNR values. All the BER values are 0.

The above experimental results shows that the proposed correlation based watermarking algorithm's performance is also stable under most common MP3 compression bit rates.

Table 5.49: An audio with various compression rate coverage tests (128 kbps)

Original Wav file	Transforming to MP3 format	BER	SNR	PSNR
32kHz, 16bit, Mono, 128kbps	128kbps	0	19.8344	45.7272
32kHz, 16bit, Stereo, 128kbps	128kbps	0	19.3149	48.0081
44.1kHz, 16bit, Mono, 128kbps	128kbps	0	22.8630	48.7731
44.1kHz, 16bit, Stereo, 128bps	128kbps	0	24.0529	52.7638
48kHz, 16bit, Mono, 128kbps	128kbps	0	22.1644	48.0701
48kHz, 16bit, Stereo, 128kbps	128kbps	0	23.4291	52.1227

Table 5.50: An audio with various compression rate coverage tests (320 kbps)

Original Wav file	Transforming to MP3 format	BER	SNR	PSNR
32kHz, 16bit, Mono, 320kbps	320kbps	0	18.9988	44.8823
32kHz, 16bit, Stereo, 320kbps	320kbps	0	19.7956	48.6978
44.1kHz, 16bit, Mono, 320kbps	320kbps	0	22.6079	48.5171
44.1kHz, 16bit, Stereo, 320kbps	320kbps	0	22.8980	51.8187
48kHz, 16bit, Mono, 320kbps	320kbps	0	21.9718	47.8757
48kHz, 16bit, Stereo, 320kbps	320kbps	0	21.4230	50.3333

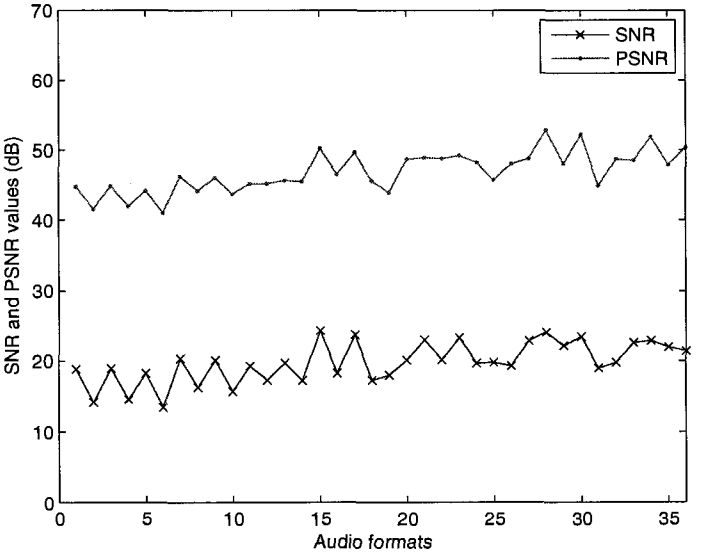

Figure 5.2: Summary of an audio with various compression rate coverage tests

5.4 Comparison and Explanation

Both Gaussian distribution based watermarking algorithm and correlation based watermarking algorithm work on MP3 audio signal directly and use original audio signal energy calculation to be a part of watermark parameters to adjust watermark coefficient adaptively. Both algorithms measure the watermarking algorithm performance by PESQMOS, SNR, PSNR and BER. Both watermarking algorithms implement blind watermark detection system.

Gaussian distribution based watermarking algorithm use Gaussian distribution anal-

ysis on both watermark embedding and detection processes. In this method, 0 or 1 watermark value is represented by a fixed watermark structure. Watermark bit 0 is represented by a "-1" fixed macro frame size watermark pattern and watermark bit 1 is represented by "+1" fixed macro frame size watermark pattern. Experimental results show this algorithm has good performance on MP3 compression and against the most common attacks. But, for the watermark data security, it is not good enough.

In order to enhance the security of the watermark data, correlation based algorithm is developed and used on watermark data during embedding and detection procedures. Correlation based watermarking algorithm support to use random watermark pattern as the watermark data. During watermark embedding, random watermark pattern correlation coefficient is calculated and correlation threshold is set. When watermark detection, correlation coefficient is used to detect the possibility of the watermark existence in a macro frames to implement blind watermark detection. Experimental results present the overall of the watermark performance is good, but the performance of against MP3 compression attack is not as good as Gaussian distribution based watermarking algorithm because correlation coefficient on original audio signal changes values more especially during MP3 compression. MP3 compression itself is a data losing procedure. After correlation coefficients impact the signal values during embedding, then MP3 compression, it bring difficulty to roll back while detection.

In summary, both watermarking algorithms are adaptive watermarking methods on MP3 audio signal. The above experimental results proved that the two proposed watermarking algorithms can be used on different kinds of music such as classical music, pop music, and speech audio, either mono or stereo, with different bit rates. Gaussian distribution based watermarking algorithm can be used in the applications which require good MP3 compression performance and correlation based watermarking algorithm can

be implemented in the applications, which require high watermark security system.

Chapter 6

Conclusions and Future Works

In this thesis, we proposed two watermarking algorithms for MP3 compressed audio signals: Gaussian distribution based adaptive watermarking algorithm and correlation based adaptive watermarking algorithm. Both watermarking algorithms embed watermark after MDCT process and before quantization process to achieve better audio quality. Human auditory system is used.

In the Gaussian distribution based adaptive watermarking algorithm, Gaussian distribution analysis and original audio energy in frames are used for adaptive control. Gaussian distribution analysis is also used on MP3 frames during watermark detection to provide blind watermark detection. Our experiments show that this algorithm provides good performance under MP3 compression and safe security against common MP3 attacks. However, the watermark data pattern in this method is fixed and thus not secure enough.

To enhance the watermark data security, we also proposed correlation based adaptive watermarking algorithm. This algorithm uses random watermark pattern in watermark embedding. Correlation coefficient and threshold are later calculated for watermark detection. Compared to Gaussian distribution based watermarking algorithm, our experiments show that the SNR and BER performance of the correlation based

algorithm is not as good as the Gaussian distribution based algorithm, especially when MP3 compression attacks are applied.

Future work includes reducing the computational cost of the watermarking algorithms. Since the Gaussian distribution analysis algorithm uses mathematical exponential expressions, the statistics analysis in this method costs heavy computation on each MP3 frame. In our experiments, for each 45-second 44.1 kHz mono MP3 audio, it takes 80 seconds for watermark embedding on Intel Pentium core 2 duo 2.80 GHz processor, 1 G RAM computer. If polynomial expressions can be used to analyze MP3 frame in stead of the exponential expressions, much of the computational time can be saved and the watermarking algorithms will be much faster.

Appendix A

ISO/IEC 11172, Part 3: Audio Table 3-B.8. Layer III scalefactor bands

These tables list the width of each scalefactor band. There are 21 bands at each sampling frequency for long (type 0,1 or 3) windows and 12 bands each for short windows.

Table A.1: Table 3-B.8a. 32 kHz sampling rate, long blocks

scalefactor band	width of band	index of start	index of end
0	4	0	3
1	4	4	7
2	4	8	11
3	4	12	15
4	4	16	19
5	4	20	23
6	6	24	29
7	6	30	35
8	8	36	43
9	10	44	53
10	12	54	65
11	16	66	81
12	20	82	101
13	24	102	125
14	30	126	155
15	38	156	193
16	46	194	239
17	56	240	295
18	68	296	363
19	84	364	447
20	102	448	549

Table A.2: Table 3-B.8a. 32 kHz sampling rate, short blocks

scalefactor band	width of band	index of start	index of end
0	4	0	3
1	4	4	7
2	4	8	11
3	4	12	15
4	6	16	21
5	8	22	29
6	12	30	41
7	16	42	57
8	20	58	77
9	26	78	103
10	34	104	137
11	42	138	179

Table A.3: Table 3-B.8b. 44.1 kHz sampling rate, long blocks

scalefactor band	width of band	index of start	index of end
0	4	0	3
1	4	4	7
2	4	8	11
3	4	12	15
4	4	16	19
5	4	20	23
6	6	24	29
7	6	30	35
8	8	36	43
9	8	44	51
10	10	52	61
11	12	62	73
12	16	74	89
13	20	90	109
14	24	110	133
15	28	134	161
16	34	162	195
17	42	196	237
18	50	238	287
19	54	288	341
20	76	342	417

Table A.4: Table 3-B.8b. 44.1 kHz sampling rate, short blocks

scalefactor band	width of band	index of start	index of end
0	4	0	3
1	4	4	7
2	4	8	11
3	4	12	15
4	6	16	21
5	8	22	29
6	10	30	39
7	12	40	51
8	14	52	65
9	18	66	83
10	22	84	105
11	30	106	135

Table A.5: Table 3-B.8c. 48 kHz sampling rate, long blocks

scalefactor band	width of band	index of start	index of end
0	4	0	3
1	4	4	7
2	4	8	11
3	4	12	15
4	4	16	19
5	4	20	23
6	6	24	29
7	6	30	35
8	6	36	41
9	8	42	49
10	10	50	59
11	12	60	71
12	16	72	87
13	18	88	105
14	22	106	127
15	28	128	155
16	34	156	189
17	40	190	229
18	46	230	275
19	54	276	329
20	54	330	383

Table A.6: Table 3-B.8c. 48 kHz sampling rate, short blocks

scalefactor band	width of band	index of start	index of end
0	4	0	3
1	4	4	7
2	4	8	11
3	4	12	15
4	6	16	21
5	6	22	27
6	10	28	37
7	12	38	49
8	14	50	63
9	16	64	79
10	20	80	99
11	26	100	125

Appendix B

ISO/IEC 11172, Part 3: Audio Table 3-C.8: Tables for converting threshold calculation partitions to scalefactor bands

The following gives the definition for cbw, bo, bu, w1, and w2.

cbw: The number of partitions (cbw) converted to one scalefactor band (excluding the first and the last partition).

bo, bu: The parameters bo and bu for converting threshold calculation partitions to scalefactor bands.

w1: The first partition which is added to the scalefactor band is weighted with w1.

w2: The last partition which is added to the scalefactor band is weighted with w2.

Table B.1: Table 3-C.8c: Sampling frequency = 32 kHz, long blocks

no.sb	cbw	bu	bo	w1	w2
0	1	0	2	1.000	0.528
1	2	2	4	0.472	0.305
2	2	4	6	0.694	0.083
3	1	6	7	0.917	0.861
4	2	7	9	0.139	0.639
5	2	9	11	0.361	0.417
6	3	11	14	0.583	0.083
7	2	14	16	0.917	0.750
8	3	16	19	0.250	0.870
9	3	19	22	0.130	0.833
10	4	22	26	0.167	0.389
11	4	26	30	0.611	0.478
12	4	30	34	0.522	0.033
13	3	34	37	0.967	0.917
14	4	37	41	0.083	0.617
15	3	41	44	0.383	0.995
16	4	44	48	0.005	0.274
17	3	48	51	0.726	0.480
18	3	51	54	0.519	0.261
19	2	54	56	0.739	0.884
20	2	56	58	0.116	1.000

Table B.2: Table 3-C.8f: Sampling frequency = 32 kHz, short blocks

no.sb	cbw	bu	bo	w1	w2
0	2	0	3	1.000	0.167
1	2	3	5	0.833	0.833
2	3	5	8	0.167	0.500
3	3	8	11	0.500	0.167
4	4	11	15	0.833	0.167
5	5	15	20	0.833	0.250
6	4	20	24	0.750	0.250
7	5	24	29	0.750	0.055
8	4	29	33	0.944	0.375
9	4	33	37	0.625	0.472
10	3	37	40	0.528	0.937
11	1	40	41	0.062	1.000

Table B.3: Table 3-C.8b: Sampling frequency = 44.1 kHz, long blocks

no.sb	cbw	bu	bo	w1	w2
0	3	0	4	1.000	0.056
1	3	4	7	0.944	0.611
2	4	7	11	0.389	0.167
3	3	11	14	0.833	0.722
4	3	14	17	0.278	0.139
5	1	17	18	0.861	0.917
6	3	18	21	0.083	0.583
7	3	21	24	0.417	0.250
8	3	24	27	0.750	0.805
9	3	27	30	0.194	0.574
10	3	30	33	0.426	0.537
11	3	33	36	0.463	0.819
12	4	36	40	0.180	0.100
13	3	40	43	0.900	0.468
14	3	43	46	0.532	0.623
15	3	46	49	0.376	0.450
16	3	49	52	0.550	0.552
17	3	52	55	0.448	0.403
18	2	55	57	0.597	0.643
19	2	57	59	0.357	0.722
20	2	59	61	0.278	0.960

Table B.4: Table 3-C.8e: Sampling frequency = 44.1 kHz, short blocks

no.sb	cbw	bu	bo	w1	w2
0	2	0	3	1.000	0.167
1	2	3	5	0.833	0.833
2	3	5	8	0.167	0.500
3	3	8	11	0.500	0.167
4	4	11	15	0.833	0.167
5	5	15	20	0.833	0.250
6	3	20	23	0.750	0.583
7	4	23	27	0.417	0.055
8	3	27	30	0.944	0.375
9	3	30	33	0.625	0.300
10	3	33	36	0.700	0.167
11	2	36	38	0.833	1.000

Table B.5: Table 3-C.8a: Sampling frequency = 48 kHz, long blocks

no.sb	cbw	bu	bo	w1	w2
0	3	0	4	1.000	0.056
1	3	4	7	0.944	0.611
2	4	7	11	0.389	0.167
3	3	11	14	0.833	0.722
4	3	14	17	0.278	0.639
5	2	17	19	0.361	0.417
6	3	19	22	0.583	0.083
7	2	22	24	0.917	0.750
8	3	24	27	0.250	0.417
9	3	27	30	0.583	0.648
10	3	30	33	0.352	0.611
11	3	33	36	0.389	0.625
12	4	36	40	0.375	0.144
13	3	40	43	0.856	0.389
14	3	43	46	0.611	0.160
15	3	46	49	0.840	0.217
16	3	49	52	0.783	0.184
17	2	52	54	0.816	0.886
18	3	54	57	0.114	0.313
19	2	57	59	0.687	0.452
20	1	59	60	0.548	0.908

Table B.6: Table 3-C.8d: Sampling frequency = 48 kHz, short blocks

no.sb	cbw	bu	bo	w1	w2
0	2	0	3	1.000	0.167
1	2	3	5	0.833	0.833
2	3	5	8	0.167	0.500
3	3	8	11	0.500	0.167
4	4	11	15	0.833	0.167
5	5	15	19	0.833	0.583
6	3	19	22	0.417	0.917
7	4	22	26	0.083	0.944
8	4	26	30	0.055	0.042
9	2	30	32	0.958	0.567
10	3	32	35	0.433	0.167
11	2	35	37	0.833	0.618

Bibliography

[1] I. J. M. International Organization for Standardization, "ISO/IEC 11172-1: Coding of Moving Pictures and Associated Audio for Digital Storage Media, Part 3: Audio," 1993.

[2] J. Chen, "MPEG Audio Codec Design using High Frequency Reconstruction and Its System Implementation," in *Proceedings of the IEEE International Symposium on Signal Processing and Its Applications (ISSPA)*, vol. 3, pp. 91–97, 1993.

[3] I. Cox, M. Miller, and M. Bloom, *Digital Watermarking.* Morgan Kaufmann, Second ed., 2002.

[4] Watermarking FQA. *[Online]. Available: http://www.watermarkingworld.org.*

[5] C. Podilchuk and E. Delp, "Digital Watermarking: Algorithms and Applications," *IEEE Signal Processing Magazine*, vol. 18, no. 4, pp. 33–46, 2001.

[6] J. Herre, J. Hilpert, and K. Linzmeier, "An Introduction to MP3 Surround," *IEEE Transactions on Speech and Audio Processing*, vol. 11, no. 6, pp. 1–9, 2003.

[7] T. Collins, "Digital Audio Coding," *[Online].*
Available: http://www.eee.bham.ac.uk/collinst, pp. 1–12, 1999.

[8] B. Erwin and M. Kesse, "MPEG 1 - Layer III Decoder, Patents IPN WO 98/1211.," *[Online]. Available: http://cegt201.bradley.edu/projects/proj1999*, pp. 1–2, 1999.

[9] D. Pan, "A Tutorial on MPEG/Audio Compression," *IEEE Transactions on Multimedia*, vol. 2, no. 2, pp. 60–74, 1995.

[10] MDCT. *[Online]. Available: http://en.wikipedia.org/wiki/MDCT.*

[11] P. Raffy, M. Antonini, and M. Barlaud, "Distortion-rate Models for Entropy-coded Lattice Vector Quantization," *IEEE Transactions on Image Processing*, vol. 9, no. 12, pp. 2006–2017, 2000.

[12] D. Kim and A. Tarraf, "Perceptual Model for Non-intrusive Speech Quality Assessment," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 3, pp. 1060–1063, May 2004.

[13] S. You and W. Chen, "A Constant-time Quantization Strategy for a Real-time MP3 Encoder," in *Proceedings of the IEEE International Symposium on Consumer Electronics*, vol. 14, pp. 59–63, 2005.

[14] S. Hacker, *MP3: The Definitive Guide.* O'Reilly Media, First ed., 2000.

[15] R. Raissi, "The Theory Behind MP3," *[Online]. Available: http://www.mp3-tech.org*, pp. 1–45, 2002.

[16] C. Xu, J. Wu, and Q. Sun, "Digital Audio Watermarking and Its Application in Multimedia Database," in *Proceedings of the Fifth International Symposium on Signal Processing and Its Applications (ISSPA)*, vol. 1, pp. 91–94, 1999.

[17] P. Bassia, I. Pitas, and N. Nikolaidis, "Robust Audio Watermarking in the Time Domain," *IEEE Transactions on Multimedia*, vol. 3, no. 2, pp. 232–241, 2001.

[18] 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 (ICMCS)*, vol. 17, pp. 473–480, June 1996.

[19] PSNR. *[Online]. Available: http://en.wikipedia.org/wiki/Peak-signal-to-noise-ratio.*

[20] PESQ and PESQ open source code. *[Online]. Available: http://www.pesq.org/.*

[21] N. Fates and F. Petitcolas, "Watermarking Scheme Evaluation Tool," in *Proceedings of the IEEE International Symposium on Multimedia Software Engineering*, pp. 328–331, December 2000.

[22] S. Craver, M. Wu, and B. Liu, "What Can We Reasonably Expect From Watermarks," in *IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*, pp. 223–226, October 2001.

[23] X. Wang, "Digital Rights Management for Broadband Content Distribution," in *Proceedings of IEEE Symposium on Applications and the Internet*, vol. 27, pp. 1–4, January 2003.

[24] SDMI. *[Online]. Available: http://en.wikipedia.org/wiki/Secure-Digital-Music-Initiative.*

[25] H. Zhao, "Watermark Attacks," in *Proceedings of the IEEE International Conference on Multimedia Communication and Information Security*, vol. 2, pp. 617–622, 2002.

[26] D. Gruhl, A. Lu, and W. Bender, "Echo Hiding," in *Proceedings of the First International Workshop on Information Hiding*, vol. 1174, pp. 293 – 315, 1996.

[27] R. Tu and J. Zhao, "A Novel Semi-fragile Audio Watermarking Scheme," in *Proceedings of the Second IEEE International Workshop on Haptic, Audio and Visual Environments and Their Applications*, pp. 89–94, Septermber 2003.

[28] I. Cox, J. Kilian, F. Leighton, and T. Shamoon, "A Secure, Robust Watermark for Multimedia," in *Proceedings of the First ACM International Workshop on Informaion Hiding*, vol. 1174, pp. 185–206, 1996.

[29] R. Pickholtz, D. Schilling, and L. Milstein, "Theory of Spread Spectrum Communications - A Tutorial," *IEEE Transactions on Communications*, vol. 30, no. 5, pp. 855–884, 1982.

[30] B. Ko, R. Nishimura, and Y. Suzuki, "Time-spread Echo Method for Digital Audio Watermarking," *IEEE Transactions on Multimedia*, vol. 7, no. 2, pp. 212–221, 2005.

[31] F. Petitcolas, *Information Hiding Techniques for Steganography and Digital Watermarking.* Artech House, First ed., 1999.

[32] Wavelet. *[Online]. Available: http://en.wikipedia.org/wiki/Wavelet.*

[33] 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, 2007.

[34] K. Kyriakopoulos and D. Parish, "Wavelet Compression Techniques For Computer Network Measurements," in *Proceedings of the Fourth ACM International Conference: Signal Processing, Pattern Recognition, and Applications*, vol. 1, pp. 109–115, 2007.

[35] L. Cai and J. Zhao, "Evaluation of Speech Quality using Digital Watermarking," in *Proceedings of the IEEE Conference on Instrumentation and Measurement Technology (IMTC)*, vol. 1, pp. 726–731, May 2005.

[36] R. W. amd P. Chai, "A New Adaptive Audio Watermarking Algorithm for Copyright Protection," in *Proceedings of the IEEE Conference on Natural Language Processing and Knowledge Engineering*, pp. 281–286, October 2003.

[37] K. Kaabneh and A. Youssef, "Muteness-based Audio Watermarking Technique," in *Proceedings of the Twenty-first IEEE International Conference on Distributed Computing Systems Workshops*, vol. 4, pp. 379–383, April 2001.

[38] S. Cheng, H. Yu, and Z. Xiong, "Enhanced Spread Spectrum Watermarking of MPEG-2 AAC," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4, pp. 3728–3731, 2002.

[39] R. Tachibanao, "Two-dimensional Audio Watermark for MPEG AAC Audio," in *Proceedings of the Fifth IEEE International Conference on Security, Steganography and Watermarking of Multimedia*, vol. 5306, pp. 139–150, June 2004.

[40] R. Ni and Q. Ruan, "Adaptive Watermarking Model and Detection Performance Analysis," in *Proceedings of the First IEEE International Conference on Innovative Computing, Information and Control (ICICIC)*, vol. 3, pp. 479–482, August 2006.

[41] G. Wu, Y. Zhuang, F. Wu, and Y. Pan, "Self-adaptive MPEG Video Watermarking Based on Block Perceptual Features," in *Proceedings of the IEEE International Conference on Machine Learning and Cybernetics*, vol. 6, pp. 3871–3876, August 2004.

[42] N. Thorwirth and J. Zhao, "Security Methods for MP3 Music Delivery," in *Proceedings of the Thirty-fourth Asilomar Conference on Signals, Systems and Computers*, vol. 2, pp. 1831–1835, October 2000.

[43] C. Wang, T. Chen, and W. Chao, "A New Audio Watermarking Based on Modified Discrete Cosine Transform of MPEG/Audio Layer III," in *Proceedings of the IEEE International Conference on Networking, Sensing and Control*, vol. 2, pp. 984–989, 2004.

[44] V. Nikolajevic and G. Fettweis, "Improved Implementation of MDCT in MP3 Audio Coding," in *Proceedings of the Tenth IEEE Asia-Pacific Conference on Multi-dimensional Mobile Communications*, vol. 1, pp. 309–312, August 2004.

[45] R. Bohme and A. Westfeld, "Statistical Characterization of MP3 Encoders for Steganalysis," in *Proceedings of the IEEE International Conference on Multimedia and Security*, vol. 3, pp. 25–34, April 2004.

[46] Gaussian Distribution. *[Online].*
Available: http://en.wikipedia.org/wiki/Gaussian-function.

[47] Skewness. *[Online]. Available: http://en.wikipedia.org/wiki/Skewness.*

[48] Outlier. *[Online]. Available: http://en.wikipedia.org/wiki/Outlier.*

[49] Correlation. *[Online]. Available: http://en.wikipedia.org/wiki/Correlation.*