The University of Osaka Institutional Knowledge Archive

Title	Audio Processing Preserving Acoustic Naturalness
Author(s)	Tachibana, Ryuki
Citation	大阪大学, 2007, 博士論文
Version Type	VoR
URL	https://hdl.handle.net/11094/215
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

The University of Osaka

Audio Processing Preserving Acoustic Naturalness

人にとって自然な音響処理に関する研究

Ryuki Tachibana

Department of Electrical, Electronic and Information Engineering

Graduate School of Engineering

Osaka University

Japan

July 2007

Acknowledgements

The research described in this dissertation was accomplished as the results of several research projects in which I have been involved at the Tokyo Research Laboratory (TRL) of IBM Japan and in the Graduate School of Osaka University. Since there are many other people who were involved in these projects, I want to thank all of them for their time, efforts, and cooperation which eventually resulted in this work.

First of all, I am exceedingly grateful to my supervisor, Professor Noboru Babaguchi of the Graduate School of Engineering of Osaka University for providing me the wonderful opportunity to study as a PhD student in his laboratory. I was only able to start work on this dissertation because he assisted me in gaining admission to the graduate school. He has guided me through my research and constructively supervised this dissertation.

I wish to thank the members of my committee – Professor Noboru Babaguchi, chairman of the committee, Professor Zen Kawasaki of the Graduate School of Engineering, Osaka University, and Professor Riichiro Mizoguchi of the Institute of Scientific and Industrial Research, Osaka University, who have given me many insightful suggestions for the dissertation.

I also take pleasure in thanking Professors Kyo Inoue, Ken-ichi Kitayama, Shozo Komaki, Seiichi Sampei, and Tetsuya Takine of Department of Information and Communications Technology of the Graduate School of Engineering, Osaka University, and Professor Takashi Washio of the Institute of Scientific and Industrial Research, Osaka University for providing thoughtful comments on the dissertation.

For text-to-speech synthesis, the dissertation is primarily based on a research project undertaken with Dr. Masafumi Nishimura and Mr. Tohru Nagano at TRL. I am especially grateful to Mr. Tohru Nagano for his valuable contributions to that research. Thanks to the understanding and support of Dr. Masafumi Nishimura, who is the technical leader of our group, and Ms. Mayumi Itakura, who is the manager of our group, I was able to complete this work successfully.

For audio watermarking, most of the research described in the dissertation was performed while I was a member of the DataHiding project at TRL. The research for Audio DataHiding was performed by a sub-group consisting of Mr. Seiji Kobayashi, Dr. Taiga Nakamura, Mr. Shuichi Shimizu, and myself. The dedication of the team members made possible the high technical level of the dissertation. In addition, the research was strengthened by challenging pilot experiments that the managers of the project, Dr. Akio Koide, Mr. Norishige Morimoto, and Mr. Masato Kurokawa, asked the group to conduct. Ms. Chiho Hirasawa, a professional pianist whom we asked for assistance, greatly contributed to the improvement of the acoustic quality through repetitive and careful listening tests.

Mr. Shannon Jacobs attached to the Human Resources Department, IBM Japan, reviewed my English in this dissertation and in many of the original papers, and gave me valuable comments not only about the English but also about the organization of the presentations.

I owe a special debt to Professor Tomio Echigo of Department of Engineering Informatics, Osaka Electro-Communication University, formerly a member of my research group, initiated the opportunity to me to start the work leading to this dissertation by introducing me to Professor Babaguchi.

I also have had the benefit of interacting with the members of the Babaguchi Laboratory during my doctoral program. I especially thank Assistant Professors Yoshimichi Ito and Naoko Nitta, and Specially Appointed Researcher, Dr. Xiaoyi Yu of the Graduate School of Engineering, Osaka University, who provided me with helpful comments and invaluable advice. The fruitful discussions with Mr. Yuta Nakashima supported my research and led to new developments in audio watermarking. I also want to thank Ms. Aya Inoue. Her skilled administrative support made my work a lot easier, although it was not always an easy task to support an off-campus student.

The research groups to which I have belonged have also been supported by many other people such as development teams, sales teams, and even people outside of the company who were involved in pilot experiments. I would like to gratefully acknowledge the support of all of these people. To avoid slighting many of them, I can only thank them collectively.

Part of this work was supported by a scholarship program of IBM Japan.

I am especially grateful to my parents who raised me to be motivated for such high objectives.

Finally I would like to acknowledge my indebtedness to my family for their endless love, understanding, support, encouragement, and sacrifice throughout my studies and work.

Dedicated to my wife Satomi,
and my children Momoha and Youji

- Ryuki Tachibana

Preface

This dissertation presents my research work on audio processing technologies in the Graduate School of Osaka University and at the Tokyo Research Laboratory of IBM Japan. The dissertation is organized as follows:

Chapter 1, the introduction of the dissertation, clarifies the purposes by providing background and summarizing the trends of the relevant technical fields.

Chapter 2 outlines the two audio processing technologies, text-to-speech synthesis and audio watermarking, that we focus on in the dissertation. Based on various usage scenarios involving these technologies in which they are used as components of larger systems, we illustrate the requirements for the technologies. In addition, we clarify the characteristics of our approach to the problems by comparing it with the related work in the research areas.

Chapter 3 describes a totally trainable text-to-speech (TTS) system, every component of which can be automatically built from human speech alone. Although total trainability is a difficult challenge, training all of the components from speech is expected to improve the naturalness of the synthetic voices as well as to reduce the building costs. Since the training of the TTS components requires various kinds of linguistic and acoustic information in addition to the speech itself, the system incrementally collects the information by combining acoustic processing and linguistic processing. For collection of the orthographic and phonemic transcriptions, the accuracies of multiple configurations for automatic speech recognition are compared. The most likely part-of-speech sequences for the recognized spellings and phonemes are calculated by a text processing module. For prosodic labels, which are important for reproduction of the speaker's characteristics, the labeling accuracy is improved by combining acoustic and linguistic models, using speaker-dependent and speaker-independent models. The accuracies of the sub-modules of the system were examined by experiments.

Chapter 4 describes a robust audio watermarking algorithm that preserves the naturalness of the sound while making it possible to detect the watermark even after the sound quality is degraded. Since conventional time-domain spread spectrum watermarking algorithms require strict synchronization of the watermark signal, their robustness against attacks that displace the watermark signal was problematic. Therefore, we introduced a robust audio watermarking method that has advantages in both robustness and acoustic quality by modifying the magnitudes of the sound according to a two-dimensional pseudo-random array (PRA) defined in the time-frequency domain of the sound. In addition, the use of multiple stretched PRAs in the detection algorithm further improves the robustness against geometric distortions of the sound without requiring too much additional computational time. The communication capacity of the algorithm is also analyzed in the last part of the chapter.

Chapter 5 extends the applications of the audio watermarking algorithm to a broader range of situations, while conventional audio watermarking research assumes limited uses of digital audio data. First, we show that the algorithm described in Chapter 4 can be applied to compressed audio and that the embedded information can be detected whether or not the watermarked audio data is compressed. Second, we describe multiple composition methods for real-time watermark embedding for analogue audio and live performances played in auditoriums, and we point out their merits and flaws. Sonic watermarking, which is one of the composition methods, is a method that can embed watermarks into the sound in the air by making the watermark sound enter the air from the speaker and mixing it with the host sound in the air. The experimental results are shown to examine the effectiveness of the methods.

Chapter 6 presents the conclusions of the dissertation.

Contents

List of Figures

List of Tables

List of Symbols

(a) Constants

Symbol	Meaning
N_w	The number of words in a sentence
N_h	The number of preceding words to be considered
N_m	The number of morae in a prosodic phrase
N_b	The number of bits in the tile
B_H	The number of frequency subband
B_W	The number of columns in a pattern block
N_{LA}	The number of pattern blocks used for synchronization
N_{b30}	The number of pattern blocks in a 30-second period
d_l	The number of frames in a pattern block
B_A	The number of tiles in a pattern block
W_B	The number of tiles assigned for a bit
W_S	The number of tiles assigned for a synchronization signal
N	The number of PCM samples in a DFT frame
$\mathrm{kmin}[b]$	The index of the lowest frequency coefficient for the b-th subband
$\mathrm{kmax}[b]$	The index of the highest frequency coefficient for the b-th subband
$F_H[b]$	The number of frequency coefficients in the b-th subband
$r_b[b]$	The ratio for indicating the width of the b-th band
$m[j]$	The bit to be embedded as the j-th bit (0 or 1)
$L[j]$	The set of tile positions, (t, b), assigned for the j-th bit
L_s	The set of tile positions assigned for the synchronization signal
$C_m[f]$	The modulus operator defined for $(0 \leq f < 4)$
$\omega_F[t, b]$	The pseudo random value for the tile at (t, b) (± 1)
T_{wm}	The threshold for the bit strengths to check existence of a watermark
T_{wk}	The threshold for the bit strengths to count weak bits
r_p	The pitch shifting rate
r_d	The ratio of degradation
r_t	The ratio of expanding the pattern block
r_f	The ratio of expanding the pattern block
γ	The coefficient by which the magnitude change is diminished by windowing
ρ_B	The correlation coefficient of the adjacent logarithmic magnitude components
ρ_L	The correlation coefficient of the logarithmic magnitudes in adjacent frames
ξ	The coefficient by which the variance of the magnitudes is increased by the windowing function

(b) Functions

Symbol	Meaning
$\mathrm{DFT}(x)$	The Discrete Fourier Transform operation of the time-domain signal x
$\mathrm{IDFT}(c_c)$	The Inverse Discrete Fourier Transform operation
	of the complex frequency components c_c
$f(x)$	The probability density function of a stochastic signal x
$Q(x)$	The cumulative distribution function of a stochastic signal x
$f_{gauss}(x)$	The probability density function of a gaussian distribution
$\mathbf{E}(x)$	The expected value of a stochastic signal x
$\mathrm{Var}(x)$	The variance of a stochastic signal x
$f_{win}(n)$	The windowing function in the time domain

(c) Variables

Symbol	Meaning
α	The accent of a mora
$w[i]$	The spelling of the i-th word
$p[i]$	The part-of-speech of the i-th word
$\eta_s[i]$	The phonemes of the i-th word
$\alpha_s[i]$	The mora accents of the i-th word
$b_e[i]$	The presence of a prosodic phrase boundary just after the i-th word
$e_g[i]$	The indicator on whether the i-th segment is problematic
$\phi[i]$	The quadruplet of the i-th word
$\psi[i]$	The triplet of the i-th word
$\mathbf{P}$	A sequence of POS labels
$\mathbf{W}$	A sequence of word spells
$\mathbf{A}$	A sequence of accents
$\mathbf{H}$	A sequence of phonemes
$\mathbf{V}$	A sequence of acoustic feature vectors
$\mathbf{B}_e$	A sequence of PP boundary presence
Φ	A sequence of quadruplet
Ψ	A sequence of triplet
$\mathbf{D}$	A sequence of duration likelihood metrics
$\mathbf{R}$	A sequence of acoustic likelihood metrics
$\mathbf{E}_g$	A sequence of problematic segment indicators e_g
g	Pitch gradient
N_d	The total number of detected entities
N_a	The number of correct entities
N_s	The number of correctly detected entities
m_d	A duration likelihood metrics
m_c	An acoustic likelihod metrics

(d) Signals and variables derived from signals

Symbol	Meaning
$x[f,n]$	The n-th PCM sample for the f-th frame of the watermarked signal
$x'[f,n]$	The n-th PCM sample for the f-th frame of the host signal
$\omega_T[f,n]$	The n-th PCM sample of the watermark signal after overlapping
$\omega_{Tf}[f,n]$	The n-th PCM sample of the watermark signal before overlapping
$c_c[f,k]$	The k-th comprex Fourier coefficient of the f-th frame
$c_a[f,k]$	The k-th amplitude coefficient of the f-th frame
$\widetilde{c_a}[f,k]$	The k-th logarithmic normalized amplitude coefficient of the f-th frame
$c_\phi[f,k]$	The k-th phase coefficient of the f-th frame
$c_r[f,k]$	The k-th real component of the f-th frame
$c_i[f,k]$	The k-th imaginary component of the f-th frame
$E_f[f]$	The total energy of the f-th frame
$c_m[f,k]$	The k-th MDCT coefficient of the f-th frame
$c_{m,q}[f,k]$	The k-th quantized MDCT coefficient of the f-th frame
$a_p[f,k]$	The k-th amplitude of the psychoacoustic model output for the f-th frame
β	The ratio of changing the amplitude of the host signal
thq	The absolute threshold of the psychoacoustic model
$s_{tile}[t,b]$	The sign assigned for the tile at (t,b)
$s_{bin}[f,k]$	The sign indicating whether the k-th amplitude of the f-th frame should be increased or decreased
$z_c[f,k]$	The k-th complex Fourier coefficient of the watermark signal
$z_a[f,k]$	The k-th amplitude coefficient of the watermark signal for the f-th frame
$z_r[f,k]$	The k-th real component of the watermark signal for the f-th frame
$z_i[f,k]$	The k-th imaginary component of the watermark signal for the f-th frame
$u[f,b]$	The amplitude of the b-th band of the f-th frame
$u_d[t,b]$	The tile value for the tile at (t,b)
$\nu_F[f,k]$	The k-th complex frequency coefficient for the f-th frame of a noise signal
$\nu_{F,r}[f,k]$	The k-th real component for the f-th frame of a noise signal
$\nu_{F,i}[f,k]$	The k-th imaginary component for the f-th frame of a noise signal
$\nu[f,n]$	The n-th PCM sample for the f-th frame of the noise signal
$y[j]$	The detection strength for the j-th bit
$r[j]$	The bit received as the j-th bit (0 or 1)
$S[f]$	The synchronization strength calculated on the assumption that the f-th frame is the beginning
y_a	The accumulated detection strength
d	The ratio of the noise to the host signal

Chapter 1

Introduction

Sound is a natural and indispensable media for human beings. It provides great benefits for human communication, perception of the environment, and cultural activities. As all kinds of information are increasingly handled with on computers and via computer networks, sound is no exception. We are increasingly coming to use various digitized sounds with electronic equipment these days. Unlike transaction data that is specifically designed for computer handling, the value of digital multimedia data is not realized until that data is perceived by humans. Digital audio data must enter the air to be appreciated as audio data.

However, even when the sound of audio data reaches our ears, we cannot perceive everything in the sound. Audio data generally has high redundancy, and there is a large gap between the amount of audio data and the information we can perceive in the sound. At the same time, human ears are very sensitive to some kinds of peculiarity or unnaturalness in a sound. It is still not very clear what characteristics in the redundant data of sounds are essential to human perception. For these reasons, computer handling of audio data still continues to be a difficult task.

Some problems are caused by the fact that computer handling of audio data is still difficult even though sound is a natural and important information media for people to use in various communication channels in day-to-day situations. Although many procedures and systems are increasingly automated, the slow pace of automation of the sound-related human-computer interface sometimes results in situations where the transfer of audio data at the interface is mediated by human operators. One example is radio broadcast monitoring where human operators have to listen to on-air music to make reports on which songs are played. The slow automation of the interface is a bottleneck for the automation of the entire system and diminishes the benefits of the systems.

Therefore, the purpose of our research is to develop audio processing technologies that enable automatic exchanges of audio data between the computer world and the real world, while preserving the naturalness of the sound and the naturalness of the human-computer interaction that use sound.

There are various audio processing technologies for the human-computer interfaces. The audio processing technologies relevant to the production of sounds for presentation to people include text-to-speech synthesis, speech production, audio watermark embedding, steganography, synthesis of the sounds of musical instruments, algorithmic composition, audio rendering, and the decoders for compression technologies. The audio processing technologies relevant to reception, recognition and analysis of sounds human produced include automatic speech recognition, automatic prosody labeling, emotion recognition, speaker recognition, audio watermark detection, steganalysis, fingerprinting, sound source separation, automatic scoring, and encoders for compression technologies.

Among these technologies, we focus on text-to-speech synthesis and audio watermarking. Although these are different technologies, they have common characteristics that both use signal processing and stochastic methods to preserve the naturalness of the sounds. In addition, both are currently a focus of attention for research and are in high demand for applications as explained by the trends of relevant technical fields summarized here.

The explosive expansion of information that is provided as text is increasing the demands for text-to-speech synthesis in recent years. With the rapid progress of communication technologies, we are concurrently seeing diversification of the means to access information. We can access various kinds of online information wherever we are with mobile electronic devices such as cellular phones, personal digital assistants (PDAs), car information systems, etc. However, in situations where we use these mobile devices when we are not at a desk, we are typically unable to devote our hands and eyes exclusively to these devices. To make the interfaces of the devices easier to use, there are increasing demands for multi-modal interfaces that allow the use of speech as well as text and graphics. Meanwhile, from the perspective of the companies providing people with the information, we see the growing importance of customer relationship management due to increasing information access by the customers and tightening competition among the companies. With the growth of the awareness of the companies' brand image, the appearance of the customer-related systems such as call centers, information desks and websites is increasing in importance. Since the pressure for cost reductions is intensifying at the same time, call center automation is expected to help with these problems. For these reasons, natural and pleasing text-to-speech synthesis appropriate to represent a brand image is desired.

Regarding the trends surrounding audio watermarking, the last several years included a violent transitional period in the music industry. Before this, the sales of records and Compact Disks (CDs) were the major and stable source of earnings for the industry. However, since the ways people enjoy music are changing drastically due to the Internet and compression technologies such as MPEG1 Audio Layer 3 (MP3) and MPEG2 Advanced Audio Coding (AAC), the music industry is being subjected to strong pressures to change their business models. The industry is trying to find ways to make their businesses flexible enough to respond to the situation changes while maintaining stable sources of revenue. For example, new services such as Internet music distribution, musical ring tones for cellular phones, subscription-based music download services, and certification that websites comply with the copyrights of the content are being introduced with the aid of various technologies. Digital watermarking can embed information such as the copyright information, use condition information, or even information about the purchaser of the content into the content itself. With these kinds of information, we can build systems that automatically monitor or control the content usage, and we can develop more flexible business models since we do not need to bill before the use of the content. For these reasons, audio watermarking is an important technology for the industry.

With these strong demands for applications, we perform research on audio processing technologies, focusing on text-to-speech synthesis and audio watermarking. The ultimate goal of our research is to allow people to use natural sound-based human-computer interfaces. We think it is most important for this purpose to preserve the naturalness of the sound and the naturalness of the human-computer interaction. This is because, no matter what convenience such technologies offer, people will not willingly use these technologies if the acoustic quality is not satisfactory. At the same time, there is a trade-off among the acoustic quality and the other characteristics such as the performance, development costs, and robustness of these technologies. Hence, we cannot focus exclusively on the acoustic naturalness. We need to improve the trade-off balance points by considering these characteristics in a comprehensive manner to achieve high and practical acoustic quality. Therefore, we are following an approach that builds the technologies from the ground up by reviewing the demands of the applications. This is also a practical approach emphasizing the utility of the applications.

According to this approach, the dissertation is organized as follows. In Chapter 2, after describing some basic concepts of the audio processing technologies, we introduce various application systems of the technologies. Based on usage scenarios of these systems, we illustrate the requirements for the technologies. In addition, we clarify the characteristics of our approaches to the problems by comparing them with the related work in these research areas. Chapter 3 discusses a framework of text-to-speech synthesis systems. Some sub-modules necessary for the framework are also presented. Chapter 4 describes a new robust audio watermarking algorithm. Theoretical and experimental analysis of the method is given. Chapter 5 extends the applications of the audio watermarking algorithm to a broader range of situations. Chapter 6 concludes this dissertation and gives the future work.

Chapter 2

Requirements for Audio Processing Technologies

In this chapter, we first describe some basic concepts of each of the audio processing technologies we focus on in this dissertation, the technologies of text-to-speech synthesis (TTS) and audio watermarking. We introduce various systems that use these audio processing technologies as components. Based on usage scenarios of these systems, we illustrate the requirements for the technologies. Then we choose some of the requirements we particularly focus on in the dissertation. In the last part of the chapter, we clarify the characteristics of our approaches to the problems by comparing them with the related work in these research areas.

2.1 Systems Using Text-to-Speech Synthesis

Text-to-speech (TTS) Synthesis is a technology that converts natural language text into speech. A TTS system usually consists of three major components (Fig. 2.1): (1) a *text processing module*, (2) a *prosody prediction module*, and (3) a *speech signal generation module*. The text processing module analyzes the input text by using a stochastic language model or heuristic

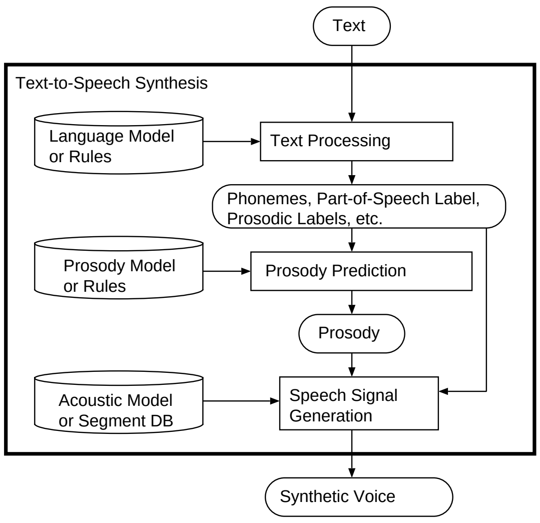

Figure 2.1: The runtime process flow of a text-to-speech synthesis (TTS) system.

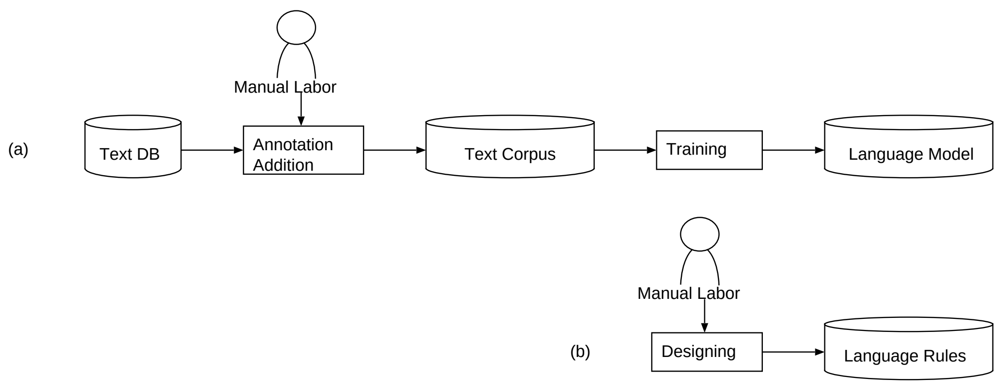

Figure 2.2: The training process flow of (a) a language model or (b) language rules.

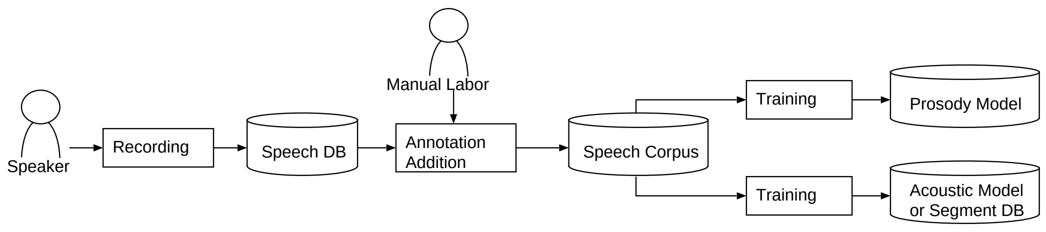

Figure 2.3: The training process flow of prosody models and a segment DB.

language rules to output linguistic information such as phonemes, part-of-speech (POS) data, prosodic labels, etc. The prosody prediction module predicts the target prosody that the output synthetic voice should have. The module uses stochastic prosody models or heuristic prosody rules. Prosody refers to the suprasegmental features such as fundamental frequencies (F0), energy, and timing of spoken language. The last component, the speech signal generation module, synthesizes the synthetic voice by referring to the target prosody and the linguistic information. In the case of concatenative TTS systems, the module concatenates speech segments stored in the speech segment database (DB).

The stochastic models and the heuristic rules have to be prepared in advance of the runtime synthesis. If the text processing module uses a stochastic language model, the model must be trained by using a text corpus (Fig. 2.2(a)). A text corpus is a large collection of text with linguistic annotation labels such as POS labels, boundary labels, prosodic labels, etc. These types of annotation labels are usually prepared manually by human labelers. If the text processing module is not based on stochastic processing, then heuristic language rules have to be prepared (Fig. 2.2(b)). The design and maintenance of the rules also require manual effort.

Prosody models, and an acoustic model or a speech segment DB are prepared based on a speech corpus, a large collection of human speech with linguistic annotation labels (Fig. 2.3). These types of annotation labels are also prepared manually by human labelers. The human speech is collected by recording the voices of a particular speaker. If the speaker has not been determined in advance, we should carefully choose a speaker whose voice is suitable for building a TTS voice. A speaker is suitable when (1) the person's voice is pleasant and intelligible for people to listen to, (2) the person can sustain the same speaking style for long recording sessions, and (3) the person's voice has good characteristics when encoded by the TTS technology. However, in some cases, it is necessary to train the models with an existing speech DB from a designated speaker. An example of such a case is when a company is using

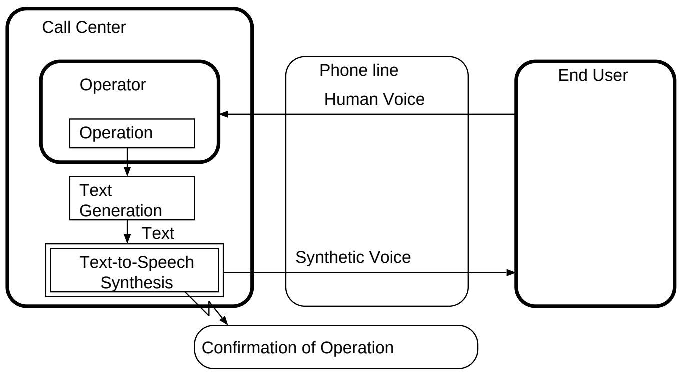

Figure 2.4: A simple computer telephony integration (CTI) system.

an old system with the recorded voice of a speaker and wants to replace that system with a new system using synthetic voices based on the same speaker. In such cases, a capability for authentic reproduction of an original speaker is desirable.

Our work is aimed at the following application systems using TTS:

- Computer telephony integration systems
- Voice browser systems
- Telematics systems

In the following sections, we describe these systems to illustrate the requirements that the systems require for their TTS components.

2.1.1 Computer telephony integration systems

In a simple computer telephony integration (CTI) system (Fig. 2.4), TTS enables confirmation of the operations. In such a system, a human operator is attending to each of the callers (end users). Each caller asks the operator to carry out an operation. When the operator types in the operation's data into the computer, the TTS system synthesizes a synthetic voice reading out the operation's data for confirmation. Both the operator and the caller can confirm the results of an operation by listening to the synthetic voices. The text to be synthesized is generated by simple rules based on the content of the operation.

Alternately, after the operator types in the operation's data and initiates speech synthesis, the operator can switch to another caller for more efficient use of the operator's time. The first caller listens to the synthetic voice to confirm the operation. If the first caller is satisfied with the results of the operation, then the caller can terminate the call. Otherwise, the caller can request assistance, perhaps by pushing a button, so that an operator will return to help fix the pending operation.

In a more complicated fully automatic interactive voice response (IVR) system (Fig. 2.5), TTS could be used to read out just the proper names or all of the information transmitted from the system to the caller. Since there is basically no operator attending the caller, the caller's speech must be recognized by using automatic speech recognition (ASR) technology. Based on the recognition results, the system automatically carries out the operations. Information such as the recognition results of the ASR, the confirmations of the operations, and the results of the operations have to be given to the callers verbally. When the information is in the form

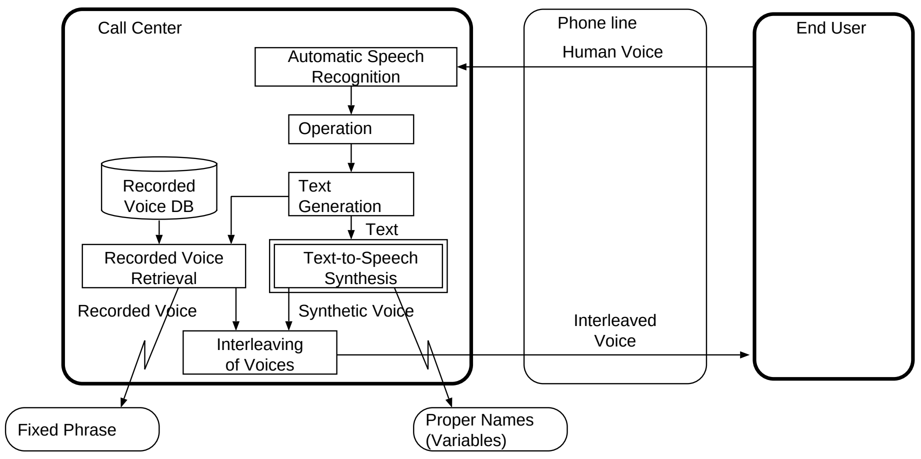

Figure 2.5: An interactive voice response (IVR) system.

of fixed phrases or when the information can be chosen from a list of prepared choices, it is common to play back recorded human voices. However, if the information includes arbitrary proper names, or if the list of responses is too long to allow preparing corresponding recorded voices, the information has to be read out with synthetic voices. In addition, while recorded voices have very good acoustic quality, it is expensive to record additional phrases when new phrases have to be added to a list of fixed phrases. In that respect, synthesizing new phrases with TTS is easy. For such reasons, a system needs to interleave recorded voices and synthetic voices.

These systems require the synthetic voices to have characteristics such as (1) intelligibility when heard through phone lines, (2) sounding natural and pleasant so that the callers are willing to continue listening during the session, (3) a proper tone for representing the company that is using the system, (4) an appropriate speaking style for the content of the spoken sentences, and (5) compatibility with the recorded voices when the synthetic voices are interleaved with recording. In addition, the following characteristics are desirable for the TTS components: (1) low computational requirements so that multiple TTS processes can run on the same IVR server in parallel, and (2) convenience when building a new TTS voice with a speaker that the company has already been using for the recorded voice.

2.1.2 Voice browser systems

A voice browser is a computer program that enables people with visual impairments to browse webpages by listening to synthetic voices reading out the pages. The end user usually controls the Web browsing by using the keyboard (Fig. 2.6). The system downloads webpages from the Internet according to the user's commands. Then the system converts each of the pages to sentences, and each of the sentences are read by the TTS component. Since webpages are usually not designed to be read aloud, the generation of sentences and synthetic voices that are easy to understand is not a simple task. In addition, while it is easy for sighted people to skip around webpages, if people with visual impairments have to listen to synthetic voices read a webpage from the top to the bottom, it can be very time consuming. Hence, real-time reading speed control and commands for skipping parts of the page are important features to enable non-visual access to webpages. Switching between different synthetic voices is a useful technique to make some parts of the webpage stand out, such as hyperlinks or itemized lists.

This kind of systems requires the TTS components to have (1) a crisp and clear enunciation

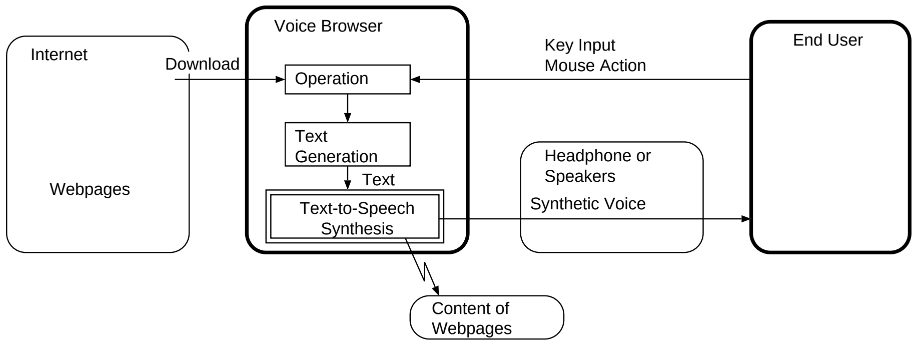

Figure 2.6: A voice browser system.

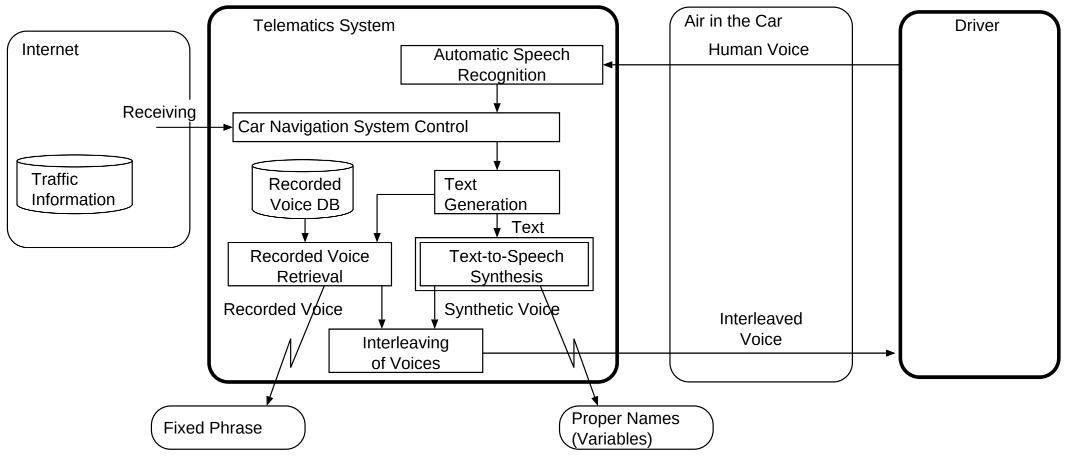

Figure 2.7: A telematics system.

for better intelligibility, (2) real-time controls, (3) easy installation for people with visual impairments, and (4) support for switching multiple synthetic voices.

2.1.3 Telematics systems

An in-vehicle telematics system is a vehicle information system that provides the driver with various information from the outside world. Because it is dangerous for drivers to look at the display of a telematics system, it is desirable that drivers can control such a system by speaking to it and that the system gives the driver all of the information verbally.

In a fully voice-enabled telematics system (Fig. 2.7), the automatic speech recognition (ASR) component of the system recognizes the commands spoken by the driver. The system decides on the required action such as setting the destination to a specified location, playing the radio, or changing the temperature setting of the air conditioner. The system should also download information by connecting to the Internet using wireless communications. The system generates sentences to be read out to return to the driver the results of the operations. In addition, even without a driver's direct command, when the system detects certain events such as approaching a turning point or receiving new traffic information, the system needs to report that information verbally. It is common that this kind of systems be equipped with a database (DB) of recorded voices reading the names of intersections, place names, and frequently used phrases. If the sentence to be read includes such prerecorded phrases,

Table 2.1: Requirements for text-to-speech synthesis. We want to address the requirements written in italic with this dissertation.

Requirements	CTI	Browser	Telematics
Intelligibility (clearness)	√	√	√
Naturalness (pleasantness)	√		√
Proper tone (elegance)	√		√
Appropriate speaking style	√		
Compatibility with recorded voices	√		√
Limited computational resource	√		√
Easy building of a new voice	√	√	√
Real-time controls		√	
Easy installation		√	
Multiple voices		√	

the system retrieves the recorded voices from the DB. When the sentence contains variable information not in the DB, the TTS component reads that information. The driver listens to the interleaved voices of the recorded voices and the synthetic voices.

Although the requirements of this kind of system are similar to the requirements of computer telephony integration systems, a peculiarity of telematics systems is that imposing a cognitive load on the driver in the act of driving could be very dangerous. Intelligibility and naturalness of the synthetic voices so that the driver need not ask again even when the spoken words are uncommon proper names is important. Another peculiarity of telematics systems is that cars are not equipped with full-function personal computers (PCs). Telematics systems must run on embedded computer systems, which have limited memory resources and computational power. Since TTS components are not a core function of these systems, the amount of available resources for the TTS components are further limited. At the same time, cars are supposed to give drivers a feeling of luxury. The synthetic voices should also have high quality appropriate for a high quality luxury environment. For example, while a sampling rate of 8 kHz is usually used for computer telephony integration systems, a high sampling rate of 22 kHz is desirable to make synthetic voices appropriate for high fidelity audio systems in luxurious cars. While the quality of synthetic voices is crucial as contributing to the appearance of the entire system, the importance of the TTS components is not well recognized, both during development and for the runtime of the entire system. Such problems make difficult the development of TTS components for telematics systems.

2.1.4 Requirements for text-to-speech synthesis

To develop a TTS technology that is attractive for owners of these kinds of systems, in this dissertation we focus on the following requirements for the TTS components of the application systems (also shown in Table 2.1):

1. **Acoustic quality and accurate reproduction of the speaker's voice**
   Although it is easy for end users listen to synthetic voices for a short time, it is difficult to make end users willingly keep listening to synthetic voices, which is a requirement of the target systems. It is necessary for TTS to generate synthetic voices that are natural and pleasant sounding, as though a living person were speaking.

2. **Rapid and accurate training of stochastic models**

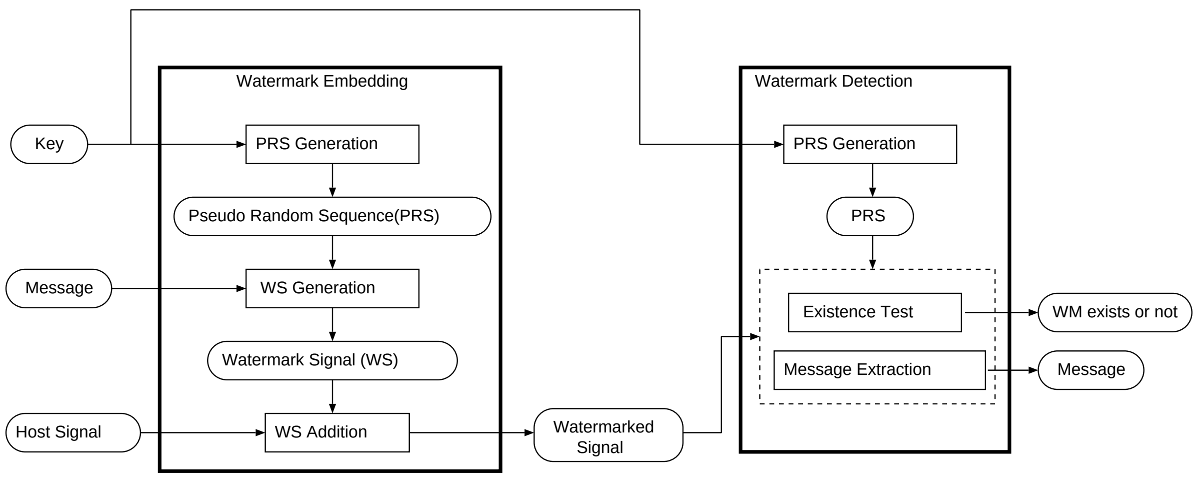

Figure 2.8: The process flow of audio watermarking embedding and detection.

> If building a new TTS voice set requires expensive manual work by skilled developers, building a different TTS voice for each of the systems would be impossible. Automatic and rapid training of stochastic models is necessary to let TTS be widely used in various systems.

Training accurate stochastic models is also important for acoustic quality, since accurate reproduction of pitch accents is crucial for natural Japanese. Hence, the focus of this dissertation is on the training stage of the models. In Section 2.3, we show what has been done with conventional approaches and what the remaining problems are.

2.2 Systems Using Audio Watermarking

Audio watermarking is a technology that allows a user of the technology to embed information into audio data by slightly modifying the data with *watermark embedding* software. The information embedded in the audio data can be detected with *watermark detection* software. Fig. 2.8 illustrates basic concepts of audio watermarking. The audio data with which the audio watermarking used is called a *host signal* (HS) or host data. The information to be embedded into the HS is a *message*. An example of a message could be copyright information about the HS. A secret *key* is used for watermark embedding. Only when the same secret key is available can the message be detected by the watermark detection software. The key is used to generate a *pseudo random sequence* (PRS). The message is encoded in a *watermark signal* (WS) based on the PRS. In the last process of watermark embedding, the WS is added to the HS, making a *watermarked signal* or a *watermarked host signal* (WHS). The watermark detection software detects the message in the watermarked signal. First, the PRS is generated from the secret key. By using the PRS, whether or not the signal is watermarked can be determined (with an *Existence Test*). If the signal appears to be watermarked, the message is extracted from the signal by using the PRS (*Message Extraction*).

We aim at the following application systems using audio watermarking:

- Pirated copy search systems
- Internet music distribution systems
- Broadcast music management systems

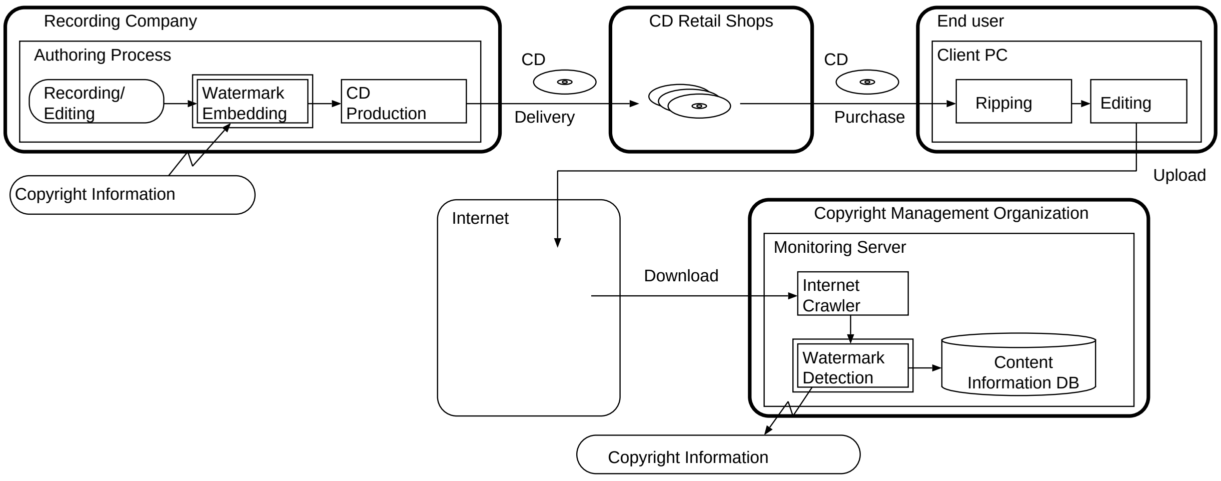

Figure 2.9: A pirated copy search system.

We describe these systems to illustrate the requirements that the systems impose for the audio watermarking components in the following sections.

2.2.1 Pirated copy search systems

In these systems, audio watermarking enables automatic searching for pirated audio files illegally made available on the Internet (Fig. 2.9). To make this possible, when audio Compact Discs (CDs) are produced at recording companies, copyright information for the audio content is embedded into the audio data. The watermarked audio CDs are distributed to CD retail shops in just the same manner as used for nomal audio CDs. End users buy the CDs from the retail shops. If the acoustic quality of the watermarked audio CDs is exactly the same as the unwatermarked audio CDs and if the watemarked CDs can be played in exactly the same manner as the unwatermarked audio CDs, the recording companies need not even publicize that the CDs are watermarked. The end users can enjoy the CDs with conventional audio equipment. The end users can even copy the CDs to analogue tapes, to their PCs, or to digital audio players. However, if the end users make the watermarked audio files available on websites or via peer-to-peer (P2P) software platforms, they will be detected and warned.

Copyright management organizations or the recording companies themselves can run Internet crawling software that searches for audio files on the Internet. By detecting audio watermarks from such audio files, the organization can know that pirated audio files have been made available illegally. Based on the copyright information detected in the audio watermark and the information about the location where the files are found, the organization can determine what action to take against the piracy, such as sending a warning, filing a complaint with the police, or beginning legal proceedings. Alternately, even without actually using these responses, simply announcing that audio CDs from the recording company have been watermarked will have a psychological deterrence effect against piracy.

The Japanese Society for Rights of Authors, Composers and Publishers (JASRAC) conducted a series of feasibility studies of audio watermarking from 1999 to 2002 targeting these kinds of systems. In 2000 and 2001, JASRAC conducted evaluation tests of audio watermark technologies provided by multiple technology providers [33, 34]. The *STEP* tests required the technologies to be used to watermark two kinds of information and then evaluated them for robustness and acoustic quality. The watermarked information was (1) 2-bit copy control

Table 2.2: Robustness requirements for the STEP test conducted by JASRAC.

Testing Item	Overview of Processing Involved
D/A, A/D transition	Digital → Analog → Digital
Altered number of channels	Stereo (2ch) → mono
Down sampling	44.1kHz/16bit/2ch → 16kHz/16bit/2ch
Amplitude compression	44.1kHz/16bit/2ch → 44.1kHz/8bit/2ch
Time and pitch compression and decompression	Time compression / decompression: ±10% Pitch shift compression / decompression: ±10%
Linear data compression	MPEG1 Audio Layer 3 (MP3): 128kbps MPEG2 Advanced Audio Coding (AAC): 128kbps ATRAC: Version 4.5 ATRAC3: 105kbps RealAudio: ISDN Windows Media Audio: ISDN
Non-linear data compression	FM (FM multiple broadcast, terrestrial hertzian TV broadcast) AM (AM broadcast) PCM (Satellite TV broadcast: communications satellite, broadcasting satellite)
Characteristic transformation of frequency response	FM (FM multiple broadcast, terrestrial hertzian TV broadcast) AM (AM broadcast) PCM (Satellite TV broadcast: communications satellite, broadcasting satellite)
Noise	White noise: S/N: - 40dB

information which should be detected from any 15-second portion of the watermarked audio files, and (2) 72-bit copyright management information which should be detected from any 30-second portion. For robustness, the test items listed in Table 2.2 were used. To test the acoustic quality of the proposed technologies, professional acoustic engineers called *Golden Ears* and *Silver Ears* evaluated the acoustic quality by listening to the watermarked audio files in a recording studio environment.

The amount of the copyright management information, 72 bits, is a number apparently intended to cover the international standard codes for this kind of information, such as the International Standard Recording Code (ISRC) and the International Standard Work Code (ISWC). An example of an ISRC is “JP-AA0-01-23456” (Fig. 2.10). The “JP” of the example is a two-character country code, which is describable with 12 bits. “AA0” is a three-character alphanumeric registrant code, uniquely identifying the organization which registered the code. This part can be described with 18 bits. The “01” is the last two digits of the year of registration, using 7 bits. The “23456” is a unique 5-digit number identifying the particular sound recording, which can be uniquely represented in 17 bits. Hence, the ISRC can be encoded with 54 bits and 72 bits is sufficient information for such copyright information.

The most crucial factor for this kind of systems is the psychological resistance of the end users and the sound engineers against watermarked audio. Regardless of whether or not the watermarked audio is distinguishable from unwatermarked audio by human ears, some people show an irrational and strong resistance to allowing audio watermarking to alter the original

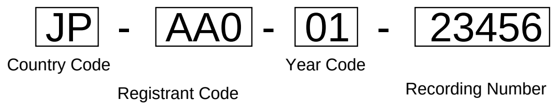

Figure 2.10: An example of International Standard Recording Code (ISRC).

audio signals. Historically, the acoustic quality of early audio watermarking technologies was not satisfactory. However, since the acoustic quality of an audio watermarking technology is heavily dependent on the watermark embedding method, rejecting all audio watermarking technology does not make sense. We need to change the attitudes of people against audio watermarking by demonstrating the quality of audio watermarking technologies.

In this use scenario, the important requirements of the audio watermarking include (1) the acoustic quality of the watermarked audio CDs should be the same as the original CDs, (2) the detection performance should allow for exhaustive Internet crawling, (3) robustness against casual editing or malicious attacks on the watermarked audio files, (4) a data payload enough large to carry the copyright information, (5) security against malicious attacks, and (6) the reliability of detected information.

2.2.2 Internet music distribution systems

In Internet music distribution systems (Fig. 2.11) in which an Internet music stores sell and distribute digital audio files to end users, audio watermarking allows not only detection of the copyright information of the audio files but also allows for the identification of malicious end users and for the direct prohibition of piracy.

First, a music store embeds the watermark into the audio files before storing the files in a music database (DB). The information embedded as the audio watermark is information identifying each audio file, such as the copyright information or usage condition information of the file. When an end user purchases a copy of an audio file, another type of watermark is embedded into the copy on the fly. This time, information that is dependent on the purchase can be embedded. For example, the identification (ID) number of the purchase or the end user can be embedded. Then the audio file is encrypted and sent to the client PC of the end user. The end user uses the proper audio player software designed for this music distribution system. The client software stores the downloaded encrypted audio file to a local music DB. When the end user wants to play the audio file, the encrypted audio file is decrypted for playback. Simultaneously, the client software automatically detects and checks the audio watermark regarding the usage conditions. If the usage of the audio file is not allowed by the usage conditions specified by the detected audio watermark, the client software stops the decryption and playback. Although this usage control mechanism can prohibit casual piracy, there is a method to bypass watermark detection. The end user could obtain an un-encrypted version of the audio file by playing the audio file, transmitting the audio signal through an analogue audio cable, and digitally recording the audio signal. After obtaining the un-encrypted version, the end user could play the un-encrypted file by using a standard digital audio player without watermark detection. However, even in this case, the watermark can still impose some risk on the malicious end user committing piracy. That is because the user ID information is still embedded in the audio file. If the audio file is found by the Internet crawling software used by the copyright management organization, the end user can be identified by detecting the audio watermark in the audio file.

In this use scenario, in addition to the requirements of Section 2.2.1, the important re-

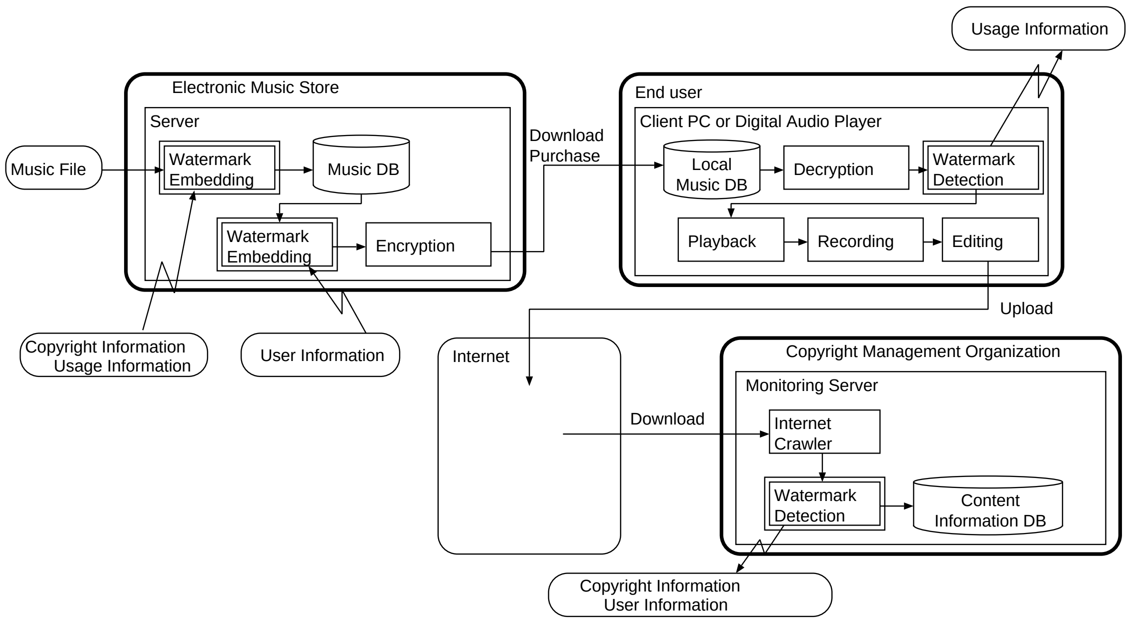

Figure 2.11: An Internet music distribution system.

quirements of audio watermarking include: (1) coexistence of multiple watermark messages, (2) embedding performance that allows watermarks to be embedded on the fly, and (3) watermark embedding for compressed audio files. The reason for the last requirement is that the compressed audio files are frequently used for Internet music distribution.

2.2.3 Broadcast music management systems

A broadcast music management system is a system that supplies radio broadcast stations with digital audio files and monitors usage of the audio files by the stations (Fig. 2.12). Audio watermarking enables automatic monitoring of the broadcast music, or, in other words, automatic generation of cue sheets by the system. A cue sheet is a report that describes the date, time, and duration of the broadcast music titles from a radio station.

In the system, the audio files are stored in a music DB owned by the music distribution service provider. The copyright information for the audio files is already watermarked in the audio files. The radio stations that are subscribing to the music distribution service can download audio files from the music DB. When a radio station requests to download an audio file, a watermark specifying which radio station is embedded into the copy of the audio file. The file is encrypted and sent to the local music DB of a client PC at the radio station. The radio station can use audio files stored in the local music DB. Engineers at the radio station can edit, play, and broadcast the audio files in the exactly same manner as standard audio files. When an audio file is transmitted over the air, the copyright management organization can automatically know which audio file is being broadcast by detecting the audio watermark. The copyright management organization owns monitoring sites equipped with monitoring equipment. The monitoring equipment includes a radio tuner, real-time watermark detection software, and a content information DB. The radio tuner is tuned to receive the broadcast from a radio station. The audio stream of the radio station is continuously fed to the real-time watermark detection software using analogue-to-digital conversion. The watermark detection software detects the watermark in real time and finds the copyright information and the station information. A

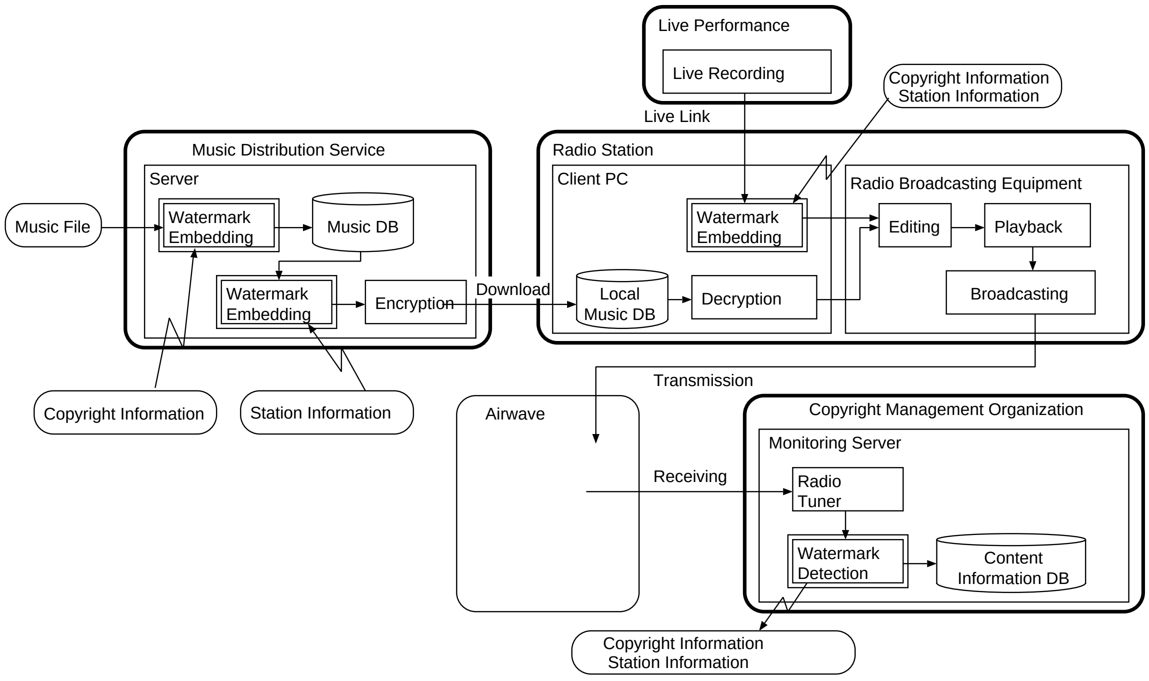

Figure 2.12: A broadcast monitoring system.

cue-sheet can be automatically generated based on the detected information for the radio station to which the radio tuner is tuned. This type of automatically generated cue-sheets will allow the copyright management organization to accurately charge royalties based on the actual usage of the audio files. Because accurate charging prevents overcharging as well as incorrect charges, the system is also beneficial to the radio stations. In addition, the copyright of live performances can also be covered by the system using real-time watermark embedding of the copyright information and station information.

In this scenario, the important requirements for the audio watermarking include (1) acoustic quality appropriate for broadcast use, (2) robustness against analogue radio broadcast effects, (3) robustness against time expansion and compression performed to adjust the duration of a song, (4) real-time watermark embedding for live performances, and (5) reliability of the real-time watermark embedding system that the engineers of radio stations can use without fear of broadcasting incidents.

2.2.4 Requirements for audio watermarking

For audio watermarking to become widely used, we want to focus on the following requirements in this dissertation among the requirements for audio watermarking in application systems (also shown in Table 2.3):

1. **Acoustic quality**

   Audio watermarking will not be widely used as long as people refuse watermark embedding software that alters the audio data, however robust the audio watermark is. Therefore, it is important to address the psychological resistance of people by developing transparent audio watermarking technologies.

2. **Robustness**

   For audio watermarking to be a useful tool for copyright management, audio watermark

Table 2.3: Requirements for audio watermarking. We want to address the requirements written in italic with this dissertation.

Requirements	Pirate search	Internet distribution	Broadcast monitoring
Acoustic quality	√	√	√
Robustness against:			
casual editing	√	√	√
malicious attacks	√	√	
analogue radio			√
duration adjustment			√
Security against malicious attacks	√	√	
Reliability of detection	√	√	
Reliability of real-time embedding			√
Embedding performance		√	√
Data payload	√	√	√
Multiple watermark messages		√	√
Compressed Audio		√	

should survive commonly used audio processing. However, in addition to malicious attacks, there are various types of audio processing in the scenarios described above that can erase or damage audio watermarks.

3. **Practical utility**
   Although audio watermarking is an interesting technology, it is not widely used. Research to expand the range of applications is also necessary to promote the adoption of audio watermarking.

Since there is a trade-off among acoustic quality, robustness, and data payload, we need to develop an algorithm that achieves a high level trade-off. In the following section, we show which problems have been solved by conventional approaches and what the remaining problems are.

2.3 Related Work

2.3.1 Related work for text-to-speech synthesis

Major approaches for TTS Major research approaches for TTS can be roughly classified into two categories: hidden Markov model-based (HMM-based) synthesis [77, 83] and concatenative synthesis [19, 28, 27]. HMM-based synthesis methods use HMMs for modeling voices and speaking patterns in a similar way to automatic speech recognition (ASR). The runtime process of HMM-based synthesis estimates the best sequence of parameter values by using the HMMs. The chosen parameter values are basically the likeliest values with respect to the models. The last stage of the method generates synthetic voices with the estimated parameter values. The method has the advantage in generating synthetic voices with stably good quality. One of the drawbacks of the method is that the synthetic voices tend to sound buzzy because of the characteristics of the filter used for voice generation. Another drawback is that the synthetic voices have a flat tone because they are generated based on the likeliest values of the parameters, which are from the mean values of many voices.

Concatenative text-to-speech (CTTS) synthesis is an approach that generates synthetic voices by concatenating speech segments of human voices. Prior to the runtime process, the training process of CTTS builds a speech segment database (DB) by separating recorded human voices into speech segments by aligning phonemic labels. The approach has an advantage in its high quality when synthesizing text that is similar to the text recorded in the speech DB. For example, the quality of a synthetic voice speaking a fixed phrase that is stored in the DB will be as perfect as the original human voice. However, a drawback of the approach is the unstable acoustic quality when synthesizing text that is too different from the recorded text. In addition, the approach requires a larger speech corpus for training models and building a DB compared to HMM-based synthesis. Both HMM-based synthesis and CTTS have drawbacks and advantages. The approach we deal with in this dissertation is the CTTS approach.

Automatic building of a new voice There is a requirement for TTS to generate new voices automatically without manual intervention. Except for the text processing module, the automatic generation of a new voice is possible even with conventional systems [19, 28, 27]. However, most conventional research work on trainable concatenative TTS systems assumes orthographic transcription of the speech. A typical procedure for building a new voice for this kind of research is as follows. The first step is the use of a text processing module for analyzing the orthographic transcription to obtain linguistic information including a phonemic transcription [28]. The method proposed by [19] alternately chooses the acoustically most likely pronunciations for each of the words in the orthographic transcription. Then the forced alignment mode of the ASR finds the phonemic alignment points that separate the speech into speech segments. The speech segments are classified by a decision tree considering the phonemic contexts of the segments and stored in the segment DB. The last step of building a new voice is training prosody models based on training data that is combination of the linguistic information, the alignment points, and the fundamental frequency (F0) values. The runtime process of the method generates synthetic voices by concatenating speech segments selected from the segment DB by referring to the target prosody predicted with the prosody models. A possible problem with the building procedure is that it requires an orthographic transcription. In some cases, we cannot assume that a correct orthographic transcription is available, since the reading script may have been lost leaving only the recordings or because the narrator might not read the script accurately. Another possible problem is that the speaker does not necessarily speak as the text processing module predicts. The prosody of the speaker can be different from the prosodic labels predicted by the text processing module. Even the pronunciations (phonemic labels) of the speaker may be different from the pronunciations predicted by the text processing module. For reasons such as these, the training of the prosody models and the segment DB with the linguistic information output by the text processing module may be less than fully accurate. Therefore, we are working towards a totally trainable TTS system taking advantage of [20]. In the framework, all of the linguistic and acoustic information that is necessary for training all of the stochastic TTS modules can be obtained almost automatically from the speech alone.

Problematic segment detection Unlike the approaches [19, 28, 27] based on the orthographic transcription, the method proposed by Adell et al. [2] automatically transcribes the speech by using ASR. When ASR is used to transcribe the speech, recognition errors cause incorrect phonemic labels. In addition, even if the phonemic labels are correctly given, alignment errors sometimes occur resulting in displaced phonemic labels. We call speech segments with these types of errors *problematic speech segments.* If the runtime process of the CTTS uses problematic speech segments for concatenation, the resulting synthetic voices will sound completely different, which is a critical error for the TTS. Therefore, it is necessary to automatically detect problematic speech segments and remove them from the segment DB. Most research [19, 40, 38] in this field uses duration-related features. The reason is that unusually short or unusually long units are most likely mislabeled. These duration-based approaches tend to improve the detection of problematic segments at the expense of precision. However, we think the precision is also important for large CTTS systems. This is because falsely labeled problematic segments dispersed in the original speech hinder the extraction of contiguous segments, but the longer segments are especially useful to make a synthetic voice of high quality. To improve both the detection rate and the precision, we use both duration-related features and acoustic likelihoods in combination.

Automatic Prosody Labeling Automatic prosodic labeling is a necessary and important component for automatic building of a new voice. An early project by Wightman et al. [81] is a foundation of this research area. This work calculated the likelihood of a prosodic label by using a decision tree based on various acoustic features. It also proposed using a Viterbi algorithm to obtain a consistent sequence of labels. After a report by Conkie et al. [14] proved that linguistic information can improve the accuracy of the labels, most research work on English prosody labeling started using linguistic information. That research also used Bayesian decisions to combine the acoustic models and linguistic models. Chen et al. [12] and Ananthakrishnan et al. [3] also proposed methods to combine an acoustic model and a linguistic model. A recent work by Lai et al. [42] reported an accuracy of 94.1% for English stress detection by using a hierarchical approach in the first stage of which linguistic information was used to distinguish between content words and function words. These projects reported POS was effective as linguistic information. The reason why the rather simple use of POS is effective for English stress detection is in the characteristics of the stress of English words. For English, only the syllable with the lexical primary stress in a word normally has the possibility of being stressed among the all of the syllables of the word. Hence, the task requires only word-level judgment on whether or not the word has a stressed syllable.

However, Japanese mora accent determination requires syllable-level judgment. We cannot expect POS to be as useful as in the case of English. For example, the information that the POS of the word is a noun does not give any hint on whether the accent of its third mora is low or high. Because of the characteristics of Japanese, linguistic information has not been used for prosody labeling research for Japanese. In the research on Japanese, Nakai et al. [46] proposed a prosodic phrase (PP) boundary detection method in which the input F0 contours were matched against F0 contour templates. The method proposed by Hirose et al. [25] can simultaneously estimate the PP boundaries and the accent types of the PPs by recognizing the input F0 contours as concatenations of PPs by using HMMs. However, the recognition capability was limited to 4-mora PPs. Although the works by Iwano et al. [32] and Hirose et al. [26] extended that method to PPs with other numbers of morae, they did not distinguish between the accent types apart from the accent types 0 and 1. The F measure of the PP boundary detection of their methods was 0.80. Emoto et al. [21] reported a method having the capability of distinguishing among all of the accent types by modeling the F0 contour including the unvoiced portions with multi-space probability distribution HMMs (MSD-HMMs). They achieved an accent type accuracy of 66.1% based on a training corpus with 450 sentences. The method proposed by Campbell [11] can detect the PP boundaries and the accent types simultaneously by searching for the prosodic label sequence resulting in the best match of a synthetic voice to the input speech. The method relies on linguistic knowledge that the developer of the text processing module of the TTS system embedded in the module. The method can recognize only accent sequences in a candidate list generated by the text processing module.

For these reasons, we use an approach that uses not only POS but also linguistic probabilities for each word. The method improves the accuracy of labeling by the combinatorial use of multiple different models including acoustic models and linguistic models.

2.3.2 Related work for audio watermarking

Basic watermarking algorithm An early watermarking research report [9] introduced several audio watermarking approaches including echo hiding and spread spectrum. Among these, time-domain spread spectrum (SS) is a well-known approach that can achieve robustness against audio compression such as MPEG1 Layer 3 [29]. While the method of this kind proposed by Swanson et al. [10, 63] required access to the host signal for watermark detection, Bassia et al. [7] proposed a time-domain SS method that enables *blind detection*, which means it can detect watermark without that access. An advantage of the research work by Swanson et al. was that it took perceptual masking [84] into consideration to make the audio watermarks inaudible. They took two types of perceptual masking effects into consideration. The first type is *temporal masking* effects where a stronger masker signal makes a temporally neighboring weaker signal inaudible. The second type is *frequency masking* effects where a stronger masker frequency component makes a weaker frequency component whose frequency is close to the frequency of the masker inaudible. Among these effects, frequency masking effects are more important for audio watermark embedding. One possible problem with the method of Swanson et al. was that, since the method was a time-domain SS method, the use of the frequency masking effects was not straightforward. The method used those effects by designing a filter simulating masking effects for each analysis frame. Differing from their method, we use frequency-domain embedding for more straightforward use of a psychoacoustic model.

Another problem of time-domain SS is that it requires exact sample-wise synchronization of a pseudo-random sequence (PRS) for watermark detection. That makes it difficult to insure robustness against pitch shifting, wow-and-flutter, and *random sample cropping*, since Wu [82] pointed out that a random sample cropping attack can efficiently interfere with a watermark detection process. The brief explanation of why they cause serious damage to audio watermarks is that they are *geometric distortions* of the audio signal. When these processing steps are performed on the content, the displacement between the embedded PRS and the detection PRS makes it difficult for the detector to properly synchronize the PRSs. For instance, a random stretching attack, which is a superset of a random sample cropping attack, randomly duplicates or deletes some portions of the content, and as a consequence the beginnings of the embedded PRSs are randomly shifted, and the lengths of the embedded PRSs are also changed randomly. In this way, random stretching and wow-and-flutter cause a mismatch of the PRSs with respect to time. Pitch shifting and wow-and-flutter causes mismatches with respect to frequency, too. These processing steps change the frequency of the embedded PRS and prevent it from matching the frequency of the detection PRS.

The effects of these processing steps are similar to the effects of geometric distortions on an image watermark, which has been receiving more and more attention recently. When an image is rotated, translated, or scaled, the mis-synchronization of the PRSs becomes a similar problem. Although one of the possible methods to deal with the distortion is an exhaustive search for the original shape of the image, that requires excessive computational time. A promising approach is using characteristics that are invariant to distortions. Similarly, it is desirable for audio watermarking methods to be insensitive to distortions. To make this work with respect to the distortions along the time axis, a promising approach is embedding in magnitudes taking advantage of the shift-invariance characteristics of the magnitudes in the Fourier domain [22, 82, 24, 79]. This approach is also used for image watermarking techniques [58, 41, 43]. For insensitivity to the distortions along the frequency axis, while the method [76] achieved robustness against pitch shifting by searching for five additional sinusoids, we aim at achieving some robustness against pitch shifting without a synchronization signal. For this purpose, we choose to modify the magnitudes of all of the frequency bins in a subband according to one pseudo-random number assigned to the subband, instead of modifying each frequency component independently [4].

To effectively realize these ideas, we use a two-dimensional pseudo-random array (PRA) in the time-frequency plane of the content. The embedding algorithm modifies the magnitudes of segmented areas in the plane according to the PRA and the detection algorithm correlates the PRA and the magnitudes of the content. The two-dimensional array allows the use of a longer array, which makes the watermark robust against time-fluctuation caused by duplicate or missing audio samples. This is because loss in some portions of the array can be recovered using other portions. The watermark robustness against pitch-shifted content can be estimated based on the amount of the segmented areas that can maintain the correspondence between the detection PRA and the watermark signal embedded in the content.

Further robustness against geometric distortion Making basic watermarking algorithms robust against geometric distortions has limits. We need to use additional means to make algorithms able to survive excessive geometric distortions. An audio watermarking method [36, 37, 35] that is similar to our approach solved the problem by performing multiple correlation tests. One possible problem with the multiple correlation tests is that they may increase the false alarm rate. This is because, if watermark is searched for by calculating correlation values for multiple times, this searching step increases the chance of correlation values calculated from unwatermarked audio content accidentally going beyond the predetermined threshold. We need a method that enables watermark detection from excessively distorted audio content while preserving the false alarm rate. Therefore, we take an approach that chooses one correlation based on the strength of a synchronization signal. Because the strength of the synchronization signal and that of the message signal are independent, the false alarm rate is kept low.

Data capacity analysis A watermarking algorithm will not be practically useful if the size of the data payload is too small, even if it achieves high acoustic quality and strong robustness. The requirements for data payload is expressed in two forms: (1) to embed as much as information as possible into audio content within a limited length, or (2) to shorten the necessary length of the audio content for embedding a certain amount of information. Although it is generally possible to increase the size of the data payload by using stronger watermarking signals, this approach degrades the acoustic quality. If we try to embed too much information in a short piece of audio with too weak a watermark signal, we will not be able to provide sufficient robustness. Recently, there is increasing demand to quantify this relationship to be able to estimate the data capacity of watermarks [6, 15, 61, 5]. This kind of research work considers watermarking as communication where host signals are treated as additive noise disturbing the communication. However, the main target of this kind of research has been image watermarking. We need to theoretically analyze the data capacity of our audio watermarking algorithms.

Watermarking for compressed audio When we think about practical usability of audio watermarking, we are seeing compressed music becoming more and more popular in recent years. However, possibly because uncompressed digital audio is still used for standard audio CDs, little watermarking research has been done for compressed audio compared to uncompressed audio, although there has been a lot of watermarking research for JPEG-compressed images and MPEG-compressed video. With the increasing popularity of compressed music, there is naturally a demand for watermarking compressed audio. In other words, it should be possible to embed and detect an audio watermark in compressed audio. In addition, robustness against compression, decompression, and re-compression is desired. However, it is difficult to handle compressed audio for watermarking because the quantization and the coding used for the compression technologies allow less freedom for manipulation.

Dittmann et al. [18] proposed a watermarking method for MPEG-1 Audio Layer 2 (MP2) [29]. The method embeds an audio watermark by changing the patterns of the scale factors, and it has the good feature of a large data payload capacity. Koukopoulos et al. [39] proposed a security improvement for this method. Steinebach [62] proposed another improvement of the method in which the capacity was optimized by making decisions based on the majority of even or odd scale factors. Since the main goals of these research projects involved authentication and annotation, robustness against decompression was not considered. Although the method proposed by Qiao et al. [57] applies a spread spectrum technique to the scale factors or quantized coefficients of MP2, its robustness is not described in the paper. Neubauer et al. [51, 52, 53, 54] proposed robust audio watermarking methods for compressed audio. One of the main characteristics of their compatible family of audio watermarking methods is that the watermark can be detected in the uncompressed domain whether the watermark was embedded when the content was compressed in MPEG-1 Layer 3 (MP3) or MPEG-2 Advanced Audio Coding (AAC) [30], or when the content was uncompressed. This convenient characteristic makes it unnecessary to decompress and re-compress a compressed audio file for watermark embedding. Watermark embedding in a compressed audio file is performed as follows. The method prepares an audio watermark signal in the time domain, and converts it to the representation used in the target compression technology. To change the values of the audio, the method recovers the coefficients representing the host signal by dequantization and decoding. Then the prepared audio watermark signal is added to the coefficients in the domain. Then the modified coefficients are quantized and encoded in the bitstream. Watermark detection is done by correlating the watermark signal and the audio signal in the time domain after decompression (if needed). Cheng et al. [13] proposed a robust audio watermarking method for AAC. Its embedding algorithm sorts the recovered coefficients, and modifies the differences between neighboring pairs of the sorted coefficients. Although robustness against decompression and re-compression is reported in the paper, the detection also requires knowing the original sequencing of the sorted coefficients. Unlike these research approaches, we aim at robust blind watermark detection in both the compressed domain and the uncompressed domain regardless of the original domain where the watermark embedding was done.

Real-time watermark embedding If we think about the yet larger picture of the practical utility of audio watermarking, we need to reconsider the applications for which audio watermarking can be useful. A digital audio watermark has been proposed as a means to identify the owner or distributor of digital audio data [8, 23, 10, 63]. Proposed applications of audio watermark are copyright management, annotation, authentication, broadcast monitoring, and tamper-proofing [23]. Of the various applications, the primary driving forces for audio watermarking research have been copy control of digital music and searching for illegally copied music [60, 34]. In these usages, it is natural to consider that an original music sample, which is the target of watermark embedding, exists as a file stored digitally on a computer. For creating a watermarked Digital Versatile Disc (DVD) Audio disc, when most of the creation process including recording, mixing, mastering, etc. is finished, copying the finished song to a computer and embedding an audio watermark in the music files on the computer is possible. In the scenario of Secure Digital Music Initiative (SDMI), an audio watermark may be embedded in a music file prepared for Internet distribution.

Although it is true that these purposes have greatly encouraged audio watermarking research to date, audio watermarking is not a technology useful only for digitally stored music. Of course, music is performed, created, stored, and listened to in many different ways, and it is much more common that music is not stored as a digital file on a computer. When applying audio watermarking technology in various musical environments, *real-time watermark embedding* is a preferred approach. For example, in music mastering studios, by embedding a watermark in real time in a sound being played, the watermarked sound could be instantly checked without saving it as a file on a computer. For broadcasting of a watermarked sound, it would become possible to embed a watermark in real time and instantly broadcast the sound of a live performance being performed in a studio, or the voice of a newscaster or the voices of participants in a talk show.

Since real-time watermark embedding is a new concept, we can think about various possible methods for combining real-time watermark embedding. Each of them, having particular drawbacks and advantages, can in a different way extend the range of audio watermark applications. For example, a real-time embedding method we named *Sonic Watermarking* could be used for prevention of *bootleg recordings* of live performances. Bootleg recordings are illegal music files that have been recorded in auditoriums by unethical audience members using portable recording devices. For movies, applications of video watermarking to the digital cinema have been gathering increasing attention recently [17, 80]. One of the purposes is to prevent a *camcording attack*, which is a recording of the movie made at a theatre. However, neither digital watermarking, encryption, nor streaming can be used in *live* performances, so there has been no efficient means to protect the copyrights of live performances in the Internet era. Sonic watermarking, a totally new application of audio watermarking, can be used for this purpose. Since watermark embedding has to be performed in a very different way compared to conventional methods, we need new evaluation methods for the acoustic quality.

2.4 Concluding Remarks

In this chapter, after we described some basic concepts of the audio processing technologies, we introduced some systems using the technologies at which our work is aimed. The systems for TTS were computer telephony integration systems, voice browser systems, and telematics systems. The systems for audio watermarking were pirated copy search systems, Internet music distribution systems, and broadcast music monitoring systems. We described the role of the technologies in the systems by clarifying the relationship with the other components. This allowed us listing of the requirements for the technologies. Then we chose some of the requirements we particularly focus on in the dissertation. Acoustic quality was chosen at the top of the requirement both for TTS and audio watermarking. No matter what convenience the technologies offer, people will not widely use those technologies if the acoustic quality is not satisfactory. Because training of accurate stochastic models is important for the acoustic quality of TTS and because manual preparation of training corpora is expensive, we focused our work on TTS to accurate and automatic training of the stochastic models. For improving the acoustic quality of audio watermarking, we need to elevate the level of the trade-off among the acoustic quality, robustness, and data capacity of the watermarking algorithm.

From such a kind of perspective, we surveyed the related work in these research areas and clarified the characteristics of our approaches to the problems. The primary characteristic of our approach to TTS is in the design of our TTS system, every component of which, including the text processing module, can be automatically built from a speech corpus. We can expect the system can well reproduce the particular quality of the original speaker by making all the components based on the speaker's speech. For this purpose, we need an accurate automatic prosody labeling method and an accurate problematic segment detection method. Combinatorial use of linguistic features and acoustic features, and duration features and acoustic features is a key of our approach. The audio watermarking approach of us is characterized by the watermark embedding algorithm in the frequency domain by modifying the magnitudes of audio content referring to a psychoacoustic model and a two-dimensional pseudo-random array. This makes the watermark inaudible to human ears, and insensitive to distortions. The wide range of applications such as compressed audio and live performances is another characteristic of our research.

Chapter 3

Totally Trainable Text-to-Speech System

In this chapter, we describe a totally trainable text-to-speech (TTS) system, every component of which can be automatically built from human speech alone. Although total trainability is a difficult challenge, training all of the components from speech is expected to improve the naturalness of the synthetic voices as well as to reduce the building costs. The system will enable us to use various synthetic voices in a greater diversity of day-to-day situations. Since the training of the TTS components requires various kinds of linguistic and acoustic information in addition to the speech itself, we introduce a framework that incrementally collects the information by combining acoustic processing and linguistic processing. For collection of the orthographic and phonemic transcriptions, the accuracies of multiple configurations for automatic speech recognition are compared. The most likely part-of-speech sequences for the recognized spellings and phonemes are calculated by a text processing module. For prosodic labels, which are important for reproduction of the speaker's characteristics, the labeling accuracy is improved by combining acoustic and linguistic models, using speaker-dependent and speaker-independent models. We show the accuracies of the sub-modules of the system examined by experiments. The results of the subjective listening tests to assess the overall quality of the synthetic voices are also shown.This chapter is related to the work published in [55, 71, 72, 45, 73, 70].

3.1 Framework of Totally Trainable Text-to-Speech System

In this section, we describe the framework of a totally trainable TTS system [55], named T^{4}S, every component of which, including the text processing module, can be automatically built from a speech corpus. The system is based on the trainable TTS engine described in [20]. The system incrementally collects the information by combining acoustic processing and linguistic processing. The sub-modules used for the system are also described in the section, except that especially important sub-modules, automatic prosody labeling and problematic segment detection, are described in the Section 3.2 and Section 3.3, respectively.

3.1.1 The T^{4}S framework

(a) Brief overview of T^{4}S

The process flow of the system is briefly illustrated in Fig. 3.1. The right side of the figure is the runtime process of the system, which is carried out in the same manner as conventional TTS

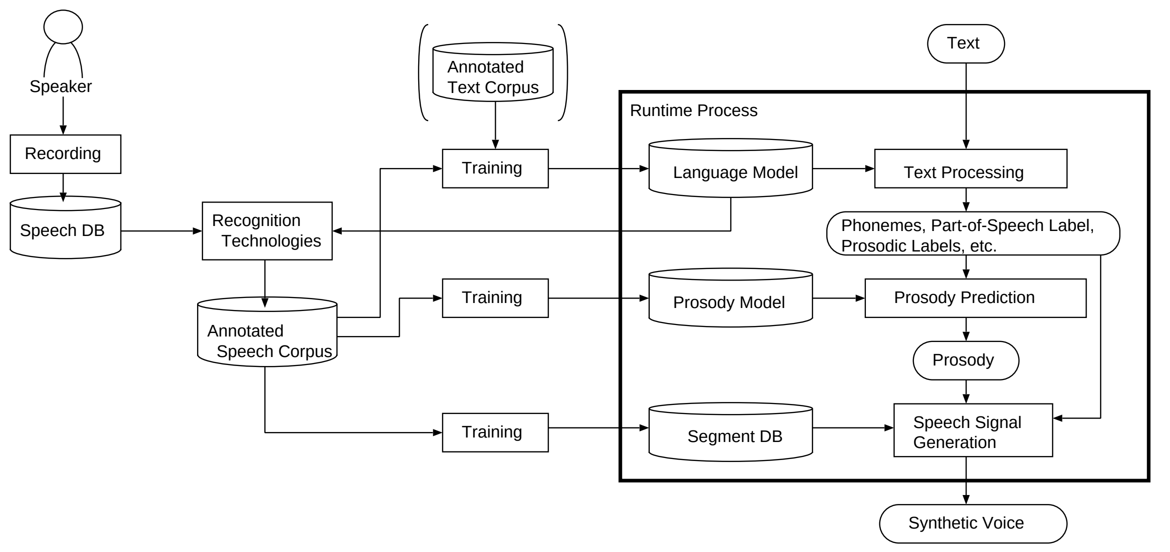

Figure 3.1: The system configuration of the T^{4}S framework.

systems (Fig. 2.1) except that T^{4}S uses a stochastic language model for the text processing module. As the research on text processing modules of TTS systems is behind the other areas of the TTS research, it is still common to use rule-based text processing modules in the society. By introducing the stochastic text processing module to the system, all of the major components of the system now become automatically trainable. The left side of the figure illustrates the training process. All of the stochastic models are trained based on an annotated speech corpus. For training the text processing module, we can additionally complement the corpus with a speaker-independent text corpus shown in the parentheses at the upper part of the figure. In this case, the speaker-independent text corpus should be adapted to the speaker of the speech corpus. The annotated speech corpus is automatically generated from the speech database (DB) by various recognition technologies instead of prepared by manual labors. The language model of the text processing module based on the speaker-independent text module can be used to augment the performance of some of the recognition technologies. The speech DB is a collection of the recorded voices of a human speaker.

(b) The training process

The training process consists of several steps to obtain acoustic information and linguistic information from the speech. The acoustic information is primarily Glottal Closure Instances (GCIs) and phonemic alignments. The linguistic information includes the orthographic transcriptions (spellings), the phonemic transcription (phonemes), the part-of-speech (POS) labels, and the prosodic labels. The process flow of the proposed framework is as follows (also shown in the shaded area of Fig. 3.2):

1. Obtain recorded speech
We do not assume that a correct orthographic transcription is available, since the reading script could be lost leaving only the recordings.

2. GCI detection
A wavelet-based tool is used to detect GCIs.

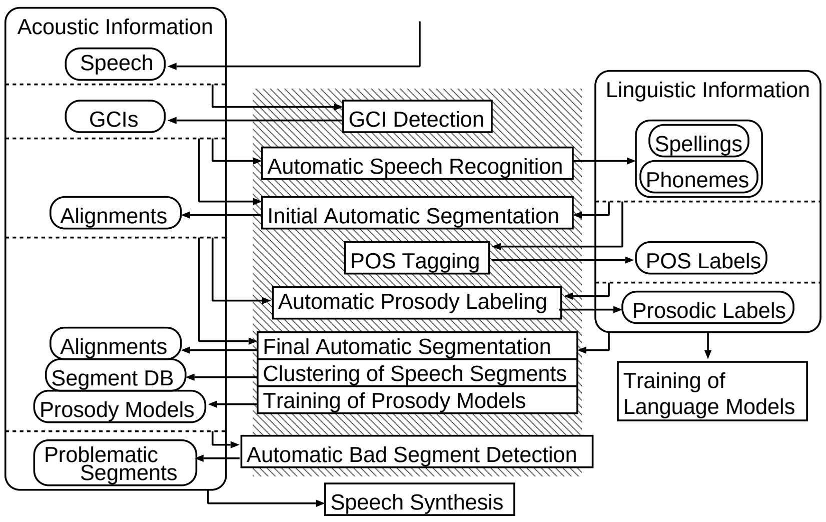
Acoustic Information
Speech
GCIs
Alignments
Alignments
Segment DB
Prosody Models
Problematic Segments
GCI Detection
Automatic Speech Recognition
Initial Automatic Segmentation
POS Tagging
Automatic Prosody Labeling
Final Automatic Segmentation
Clustering of Speech Segments
Training of Prosody Models
Automatic Bad Segment Detection
Speech Synthesis
Linguistic Information
Spellings
Phonemes
POS Labels
Prosodic Labels
Training of Language Models

Figure 3.2: The detailed process flow of the T^{4}S framework. GCI and POS stand for *Glottal Closure Instance* and *Part-Of-Speech*, respectively.

3. ASR and initial phonemic segmentation
After ASR transcribes the speech, the forced alignment mode of the ASR finds phonemic alignment points.

4. POS tagging
A text processing module calculates the most likely POS sequence for the recognized spellings and phonemes.

5. Automatic prosody labeling (Section 3.2)
This step annotates prosodic labels into the speech.

6. Final phonemic segmentation, segment clustering and prosody model training
The conventional methods [20] do these processes based on the estimated values of linguistic and acoustic information.

7. Automatic problematic segment detection (Section 3.3)
Errors in transcription and segmentation must be automatically detected and removed from the candidate list of speech segments available for speech synthesis.

8. Speech Synthesis
We use a conventional TTS engine [20] in the runtime.

3.1.2 Sub-modules of T^{4}S

We describe some sub-modules in the proposed framework and report experimental results using the sub-modules.

(a) Text processing module

The text processing module of a TTS system is the runtime module that estimates phonemes and accents for inputted text. The accuracy of the estimation is critical for generating intelligible and natural synthetic voices. As the acoustic quality of TTS systems has been greatly improved, the poor accuracy of phonemes and accents predicted by rule-based text processing modules is becoming a major issue of the TTS research. The rules of a rule-based text processing module have to be maintained by costly manual labor of skilled developers. In addition, the rule-based approach has limitations in the scalability and the ease of domain adaptation,

To tackle these problems, we introduced a stochastic text processing module [44] to T^{4}S. The stochastic-based approach has advantages in the ease of adaptation to specific domains or speakers. The n-gram-based module simultaneously solves word segmentation[1], grapheme-to-phoneme conversion, homograph disambiguation, and accent generation. We define the unit of the n-grammodel ϕ as a quadruplet of spelling of a word w, its POS p, its phoneme sequence η_s, and its accent sequence α_s, that is $\phi = \langle w, p, \eta_s, \alpha_s \rangle$. The module obtains the likeliest sequence ϕ_{max} of quadruplets for the inputted text by the following equation:

$$\phi_{max} = \underset{\phi}{\mathbf{argmax}}\, P(\phi). \tag{3.1}$$

The probability of a quadruplet sequence $\Phi = (\phi[0], \phi[1], ..., \phi[N_w - 1])$ is calculated by multiplication of the probabilities of the components:

$$P(\Phi) = \prod_{i=0}^{N_w-1} P(\phi[i] | \phi[i - N_h], .., \phi[i-2], \phi[i-1]), \tag{3.2}$$

where N_w is the assumed number of words in the text and N_h is the predetermined length of the word history used for the probability calculation. For example, the value of N_h is 1 for a bigram model. It was shown by Nagano et al. [44] that the accuracy of the stochastic text processing module trained based on a text corpus with approximately 10,000 sentences is better than that of a rule-based module.

(b) Automatic speech recognition

In the training process of T^{4}S, automatic speech recognition (ASR) is used to obtain the spellings and the phonemes of the speech. We compared the accuracies of the following transcription methods. Note that these accuracies are heavily dependent on the designs of the dictionaries and the language models.

R1 The orthographic transcription is available. The text processing module of the TTS system converts it to the phonemic transcription [28].

R2 The orthographic transcription is available. The text processing module of the TTS system splits the script into words. The acoustically most likely pronunciation of each of the words is chosen from a pronunciation dictionary [19].

R3 The orthographic transcription is unavailable. ASR with a speaker-independent acoustic model transcribes the spellings and phonemes based on the acoustic likelihood. This is our proposed method. We used the spellings of the reading script for training the language model in the experiments, because ASR generally requires a language model trained for the domain of the speech.

We conducted experiments using a speech corpus of 5,376 sentences recorded by an adult female. We manually prepared a test corpus of 200 sentences out of the whole corpus and measured the phoneme accuracies of the methods. The phoneme error rates of **R1**, **R2**, and

[1]Word segmentation is a procedure to separate text into words. The procedure is necessary for languages whose text is not separated by spaces.

R3 were 0.779%, 1.66%, and 0.997%, respectively. Although orthographic transcription was not used, the results showed that **R3** was able to achieve accuracy close to that of **R1**. The reason the accuracy of **R2** was poor is that it is susceptible to inadequate entries or extraneous entries in the pronunciation dictionary and that it leaves the word context out of consideration.

(c) POS tagging

The POS labels of the speech were obtained, based on the recognized spelling and phonemes, by using a similar technique to the stochastic text processing module [44]. The most likely POS sequence **P** for the given spelling sequence (**W**) and phoneme sequence (**H**) is the one that maximizes the posterior probability $P(\mathbf{P}|\mathbf{W},\mathbf{H})$ in the following equation:

$$P(\mathbf{P}|\mathbf{W},\mathbf{H}) \;=\; \frac{P(\mathbf{W},\mathbf{P},\mathbf{H})}{\sum\limits_{\forall \mathbf{P}} P(\mathbf{W},\mathbf{P},\mathbf{H})}, \tag{3.3}$$

where the summation of the denominator is used for all of the possible POS sequences for **W** and **H**. The joint probability can be calculated as the product of the n-gram probabilities:

$$\begin{aligned} &P(\mathbf{W},\mathbf{P},\mathbf{H}) \\ &= \prod_{i=0}^{N_w-1} P(w[i],p[i],\eta_s[i]|w[0],p[0],\eta_s[0],..,w[i-1],p[i-1],\eta_s[i-1]), \end{aligned} \tag{3.4}$$

where N_w is the number of words, and $w[i]$, $p[i]$, and $\eta_s[i]$ are the spelling, the POS, and the phonemes of the i-th word, respectively. The n-gram probability is calculated from the n-gram frequency in the speaker-independent text corpus.

We conducted an experiment to examine the accuracy of the POS tagging for automatically recognized spellings and phonemes. We measured the POS accuracy for the words as units for the same 200 sentences, and the word error rate (WER) was 4.55%. When we used the text processing modules with the correct orthographic transcription for comparison, the WER was 4.16%. Hence, the proposed method without the orthographic transcription was able to achieve the same accuracy as with the orthographic transcription. In contrast, when the POSs for words were simply looked up in the pronunciation dictionary, the WER was 43.2%. Therefore, we can see simple lookup does not work well. This is because there are some words with different POSs but sharing the same spelling and phonemes, because we are using short word units.

(d) Prosody models

The prosody models generate target prosody of the synthetic voice by using decision trees based on linguistic information estimated by the text processing module in the runtime process of TTS. The target prosody consists of the duration, the fundamental frequency (F0), and the energy of the phonemes. While the conventional system trained the prosody models based on the output of the text processing module, we train the prosody models based on the annotated speech corpus that is generated from the speech DB by using recognition technologies. This resolved inconsistency between the prosodic labels and the actual speech. Consequently, we were able to improve the prediction accuracy of the prosody models.

(e) Segment DB

The segment DB stores the speech segments by classifying the segments using decision trees. The decision trees are trained based on the information gathered by the recognition technolo-

Table 3.1: The results of the subjective listening tests. The numbers are Mean Opinion Score (MOS) values.

	Human Voice	Synthetic Voice
In-domain	4.9	4.1
Out-of-domain	4.8	3.2

gies. The conventional system used only the phonemic context of each phoneme as the input features for the decision trees. Since the automatic prosody labeling algorithm enabled more accurate labeling, we added the mora accent label to the input features so that the context of the accents is taken into consideration for segment selection.

3.1.3 Results of subjective listening tests

We conducted a subjective listening test to evaluate the overall quality of synthetic voices of T^{4}S. Although the system is designed for automatic generation of a new voice, since accumulation of slight errors in the sub-modules results in non-ignorable errors in the synthetic voices, we built the system by manually fixing the training data as needed. The system was built based on a speech DB containing 8,851-sentence for a female professional speaker. Eighteen subjects rated, in five levels, the quality of the synthetic voices by comparing them with recorded human voices. We used text that was not included in the speech DB. The means of the results were calculated for each of in-domain text and out-of-domain text (Table 3.1). In-domain text is text in a domain for which a considerable number of sentences are recorded in the speech DB. Car navigation and weather forecast are the domains used for this category. Out-of-domain text is text in a domain for which there is no recorded voices in the DB.

The results showed that the system was able to generate synthetic voices with a quite high quality for in-domain text. However, the acoustic quality for out-of-domain text certainly has a room to improve.

(a) Discussion

We introduced the T^{4}S framework, every component of which can be automatically built from a speech corpus. We showed the experimental results for the accuracy of the sub-modules. Description of especially important sub-modules, automatic prosody labeling and problematic segment detection, will be given in the following sections. The acoustic quality of the synthetic voices by the system was evaluated by the subjective listening tests although totally automatic generation was not used for the system. The results revealed that there was a room for improvement in the acoustic quality for out-of-domain text. The inadequate score for out-of-domain text was caused by inaccurate pitch contours and discontinuity at concatenation points. We need to improve the spectral continuity calculation algorithm and the segment search algorithm. We have not established a method for speaker adaptation of the language model for the text processing module. When this is done and we achieve a certain level of accuracies for the sub-modules, the overall effectiveness of the framework should be evaluated. Quality of reproduction of the speaker's speaking style by the system should also be assessed.

3.2 Automatic Prosody Labeling

Automatic prosody labeling is a task to automatically annotate prosodic labels such as syllable stresses or break indices into the speech corpus. Since errors in the prosodic labels can lead

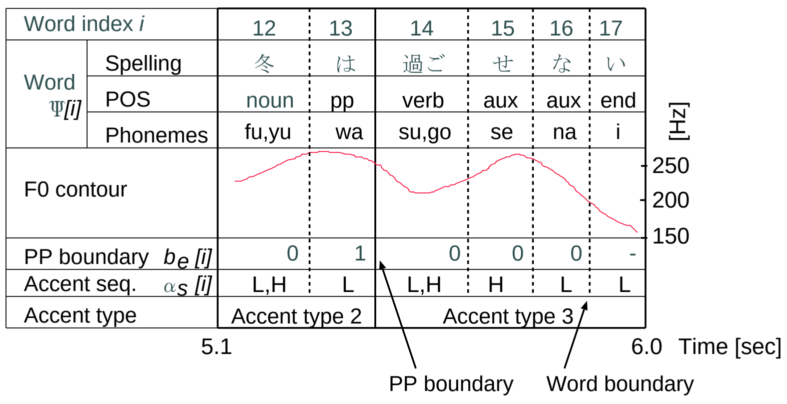

Figure 3.3: An example of Japanese utterance. POS stands for *Part-Of-Speech*. *F0* is fundamental frequency of the speech. PP stands for *Prosodic Phrase*.

to incorrect prosody estimation and unnatural synthetic sound, the accuracy of the labels is a key factor for text-to-speech (TTS) systems. In particular, mora accent labels that are pitch-related prosodic labels are very important for Japanese, since Japanese is a pitch-accent language and Japanese people have a particularly keen sense of pitch accents.

For Japanese, we need prosodic phrase (PP) boundary information and mora accent sequences as prosodic labels for speech corpora. An example of a Japanese utterance is shown in Fig. 3.3. The utterance is a part of a spoken sentence. The upper part of the figure shows the spoken words and the smoothed fundamental frequency (F0) contour, which we assume we can use for the automatic labeling. The lower part of the figure shows the PP boundary information and the mora accent sequence information that is to be automatically labeled. Although mora accent is important in Japanese speech, since differences in mora accents subtly change the physical features of the speech, accurate labeling is a difficult task.

In this chapter, we describe an automatic prosody labeling method having the following advantages:

- The combinatorial use of multiple different models including acoustic models and linguistic models improves the accuracy of labeling.
- A speaker-independent language model based on a speaker-independent text corpus represents knowledge about the possible correct accentuations in Japanese.
- Use of the *text* corpus reduces the required size of the speaker-dependent *speech* corpus (which is costly and time consuming to create).
- Searching for the best sequences of prosodic labels by using the Viterbi algorithm while considering the effects of the history of the labels improves the consistency of the labels.

3.2.1 Prosodic structure and accents of Japanese

Before introducing the proposed method, we describe the prosodic structure of Japanese and the goal of the proposed automatic accent labeling method in this section. We also give the reason why automatic labeling of Japanese is a difficult task.

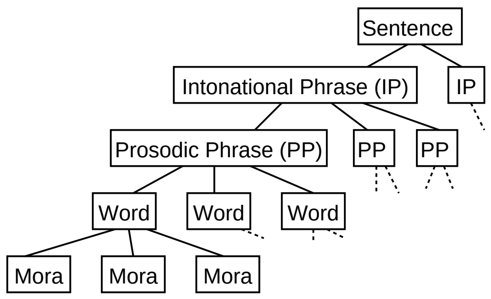

Figure 3.4: Prosodic structures of Japanese.

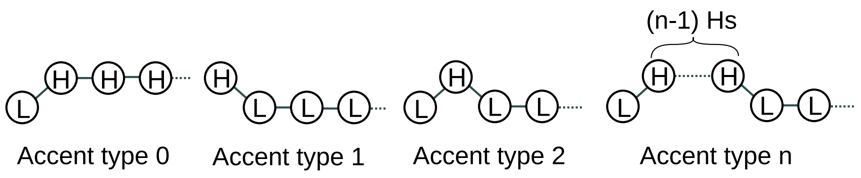

Figure 3.5: Correct accent types of Japanese.

(a) Prosodic structure

We rely on the prosodic structure of Japanese as illustrated in Fig. 3.4. That is, a sentence utterance consists of intonational phrases (IPs), which are separated by periods of silence. An IP consists of prosodic phrases (PPs). A PP is a group of words that are uttered in a prosodic combination. As shown in Fig. 3.5, the accent type of an N_m-mora PP can be either one of the accent type 0 up to the accent type $(N_m - 1)$ [2]. The phonemes in a word are grouped into morae. A mora consists of one or zero consonants and a vowel, and is a phonetic unit similar to a syllable. The first goal of the proposed method is to automatically detect the boundaries of the PPs in speech.

(b) Mora accents

The pitch accents of standard Japanese have two values, high (`H`) or low (`L`), so each mora is `H` or `L`[3]. The other goal of the proposed method is to automatically determine the accent of each mora in the speech. Although there are 2^{N_m} possible pitch patterns for a word with N_m morae, only certain patterns are actually used in Japanese (also shown in Fig. 3.5).

Accent type 0 Uniformly `H`, except for the first mora, which is `L`.

Accent type 1 Uniformly `L`, except for the first mora, which is `H`.

Accent type n ($2 \leq n < m$) It starts `L` and immediately transitions to `H`. The remaining $m - 1$ morae start `H` and end `L`, with only one transition. The number of `H` morae is $n - 1$. There are $m - 1 - 1 = m - 2$ possible locations for the second transition.

Counting all of the possibilities, the grand total is m possible patterns. Every Japanese word has one of only a few *correct* accent types, and they can be looked up in a good Japanese

[2]We do not need to consider the accent type m of PPs since the accent type 0 has the same accent sequence.

[3]We ignore secondary accent nuclei.

dictionary. The accent type for *"Kyouto"* (*"Kyoto"* in English) is the accent type 1 `(H,L,L)`. When words are uttered in a longer PP, the accent sequences of the words can change according to the context. The accent sequence of *"Kyouto"* is changed into `(L,H,H)` when it is in the PP *"Kyouto tawa- wa"* (*"Kyoto tower is"*). The accent sequence of a longer PP must still be one of the accent types listed above. The accent type of the second PP in Fig. 3.3 is the accent type 3 `(L,H,H,L,L)`. There are rules for forming a PP and for accent change, which Sagisaka et al. analyzed in [59]. An incorrectly formed PP or an incorrectly accented sequence causes listeners to misunderstand the context and syntactic structure of a text.

(c) Arbitrariness

While there are *correct* accent types, there is still some arbitrariness in the sequencing of accents. This is because (1) There are words with more than one correct type of accent; (2) Ambiguity in the meaning of a compound word can cause ambiguity in the form of the PP forming; and (3) The form of the PP is partly dependent on the speaking style and personality of the speaker. For example, when a sentence is spoken quickly, rises and falls in the accents tend to be omitted. In addition, paralinguistic information carried by the utterance affects the accent sequence. For example, when emphasis is put on a word, rises and falls in the neighboring words sometimes disappear to make the emphasized word stand out.

For these reasons, we cannot determine the accents of the spoken sentence only from the spoken text. We need to acoustically analyze the speech, although we can use the known constraints on accent sequences in Japanese.

In addition, automatic labeling of Japanese mora accents is a more difficult task than for English syllable stress in a way. For English, only the syllable with the lexical primary stress of a word normally has the possibility of being stressed among the all of the syllables of the word. Hence, the task requires only word-level judgment on whether or not the word has a stressed syllable. This is one of the reasons the part-of-speech (POS, which is word-level information) is useful for automatic labeling. However, Japanese mora accent determination requires syllable-level judgment. We cannot expect POS to be as useful as in the case of English. For example, the information that the POS of the word is a noun does not give any hint on whether the accent of its third mora is `L` or `H`.

3.2.2 Automatic prosody labeling using multiple models

To make it possible to accurately label Japanese prosodic labels, we discuss a method that uses multiple models in combination as described below.

(a) Main ideas

We split the labeling problem into two layers: (1) prosodic phrase (PP) boundary detection problem and (2) accent determination problem, and address these in this order. We use an acoustic model and a linguistic model in combination in each layer to achieve high accuracy. As there are two models for each of the two layers, there are four different stochastic models for the whole problem (Table 3.2). Training corpora are necessary for training these stochastic models. We use a speaker-dependent speech corpus for training three models among the four. This is because it is difficult to remove speaker dependency from acoustic features and because we believe PP formation is dependent on the speaker and the speaking style. However, it is desirable to minimize the required size of the corpus, since preparation of this type of corpora is costly. Hence, we employ a speaker-independent text corpus for training the linguistic accent model (bottom right of Table 3.2). By using not only POSs but also the

Table 3.2: Four models used in the proposed method. PPB stands for *Prosodic Phrase Boundary.*

	Acoustic model	Liguistic model
PPB detection	Speaker-dependent GMM	Speaker-dependent Decision tree
Accent determination	Speaker-dependent Decision tree, GMM	Speaker-independent n-gram

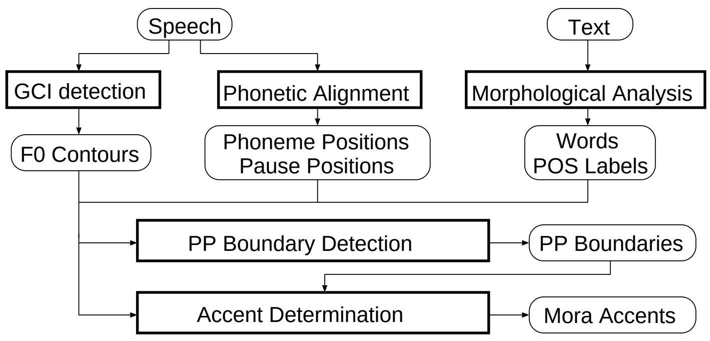

Figure 3.6: The process flow of the automatic prosody labeling method.

spellings of the words in the corpus, the corpus can serve as a rich information source about the possible correct accentuations in Japanese that are not dependent on the individual speakers. Use of the spellings for the linguistic accent model requires use of the large text corpus for training. Otherwise, for example, if we only use the small speaker-dependent speech corpus, the prediction capability of the model will be very poor because of the sparseness of the training data. For that reason, we use the large speaker-independent text corpus. In addition, a text corpus is easier to gather than a speech corpus and a speaker-independent corpus is easier to gather than a speaker-dependent corpus.

The process flow of the method is illustrated in Fig. 3.6. The minimum requirements for the method are a set of speech recordings and the corresponding text. As preprocessing for the method, glottal closure instance (GCI) detection, phonetic alignment, and morphological analysis are required. Smoothed F0 contours are calculated from the GCIs. The alignments of the phonemes are found by using an automatic speech recognition (ASR) based phoneme alignment tool. Sentence utterances are separated into IPs at the pause positions. A morphological analyzer [44] analyzes the word boundaries and the POS of each word. After these preprocessing procedures, PP boundary detection (Section (b)) is performed to separate each of the IPs into PPs. The accent type of each of the PPs is determined last (Section (c)).

(b) Prosodic phrase boundary detection

The objective in this layer is to determine the PP boundaries among all of the word boundaries in the given IP. The IP boundaries are excluded from the candidate list for the PP boundaries. We let the word sequence of the IP $\Psi = (\psi[0]\psi[1]..\psi[N_w-1])$, where $\psi[i]$ denotes the i-th word of the IP and N_w is the number of words in the IP. $\mathbf{B}_e = (b_e[0]b_e[1]..b_e[N_w-1])$ is a sequence of the locations of the PP boundaries, where $b_e[i] = 1$ denotes the presence of a PP boundary just after $\psi[i]$. The other possible value of $b_e[i]$ is 0, used if there is no PP boundary at that location. An example of the word sequence Ψ and the PP boundary sequence $\mathbf{B}_e$ is shown in

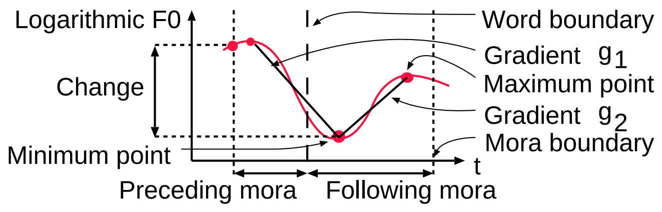

Figure 3.7: Features for the acoustic boundary model. The curve is the smoothed F0 contour. F0s on the curve are observed and used for calculation of the features.

Fig. 3.3. While $b_e[13] = 1$ means that there is a PP boundary just after the word $\psi[13]$ ("wa"), $b_e[12] = 0$ means that there is no PP boundary just after the word $\psi[12]$ ("fuyu"). The reason $b_e[17]$ is labeled '-' is that the word boundary just after the word $\psi[17]$ is excluded from the candidate list since $\psi[17]$ is the last word of the sentence and is actually an IP boundary.

$\mathbf{V} = (v[0]v[1]..v[N_w - 1])$ is a sequence of the acoustic feature vectors observed at the word boundaries. The objective of PP boundary detection can be restated as a search for the $\mathbf{B}_e$ that maximizes the posterior probability for given Ψ and $\mathbf{V}$ by using Bayes' Theorem.

$$\begin{aligned} \mathbf{B}_{emax} &= \underset{\mathbf{B}_e}{\mathbf{argmax}}\, P(\mathbf{B}_e|\Psi, \mathbf{V}) && (3.5) \\ &= \underset{\mathbf{B}_e}{\mathbf{argmax}}\, \frac{P(\mathbf{V}|\Psi, \mathbf{B}_e)P(\mathbf{B}_e|\Psi)}{P(\mathbf{V}|\Psi)} && (3.6) \\ &= \underset{\mathbf{B}_e}{\mathbf{argmax}}\, P(\mathbf{V}|\Psi, \mathbf{B}_e)P(\mathbf{B}_e|\Psi), && (3.7) \end{aligned}$$

where '**argmax**' is the operator that returns the value of the argument that maximizes the following term. $P(\mathbf{V}|\Psi)$ in Eq. (3.6) can be ignored when we only want to find $\mathbf{B}_{emax}$. $P(\mathbf{V}|\Psi, \mathbf{B}_e)$ can be obtained by using the acoustic boundary model, while $P(\mathbf{B}_e|\Psi)$ is a linguistic probability calculated by using the linguistic boundary model. Since the presence of a PP boundary at a word boundary has an effect on the neighboring word boundaries, we search for $\mathbf{B}_{emax}$ by using the Viterbi algorithm.

Acoustic boundary model We ignore Ψ in $P(\mathbf{V}|\Psi, \mathbf{B}_e)$ and approximate its value by a purely acoustic probability $P(\mathbf{V}|\mathbf{B}_e)$ focusing on the presence of the boundary. In addition, ignoring the effect of the neighborhood, we approximate the possibility of the sequence by the product of the possibilities of the elements as follows.

$$P(\mathbf{V}|\mathbf{B}_e) \simeq \prod_{i=0}^{N_w-1} P(v[i]|b_e[i]). \qquad (3.8)$$

$P(v[i]|b_e[i])$ is calculated by using multivariate Gaussian Mixture Models (GMMs) trained by using the training corpus.

The feature vector, $v[i]$, is a three dimensional vector whose components are (1) the change of the logarithmic fundamental frequency (F0) in the preceding mora, (2) the logarithmic F0 gradient in the following mora (g_2 illustrated in Fig. 3.7), and (3) the change of the logarithmic F0 gradient at the point ($g_2 - g_1$ in the figure). As shown in the figure, a minimum point near the word boundary and maximum points in the neighboring morae are searched for when calculating these features.

Although we tried to improve the accuracy of the model by adding duration-related and energy-related features to the feature vector, those kinds of features were ineffective. The

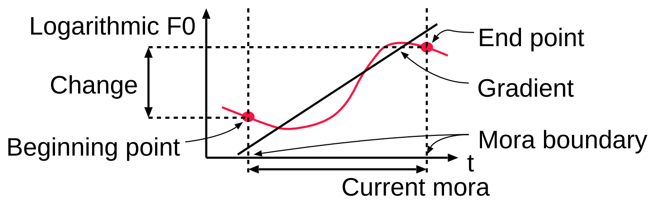

Figure 3.8: Features for the acoustic accent model.

features we tried include the durations of the preceding mora and the following mora, the ratio of the durations, the logarithmic RMS (Root Mean Square) amplitudes of the same morae, and the difference of the amplitudes.

Linguistic boundary model The linguistic probability is calculated by using the following equations.

$$P(\mathbf{B}_e|\Psi) = P(b_e[0],..,b_e[N_w-1]|\Psi) \tag{3.9}$$

$$= P(b_e[0]|\Psi)\prod_{i=1}^{N_w-2} P\left(b_e[i]|b_e[0],..,b_e[i-1],\Psi\right) \tag{3.10}$$

$$\simeq P(b_e[0]|\psi[0],\psi[1])\prod_{i=1}^{N_w-2} P\left(b_e[i]|b_e[i-1],\psi[i],\psi[i+1]\right). \tag{3.11}$$

The conditional probability $P(b_e[i]|b_e[i-1],\psi[i],\psi[i+1])$ is calculated from a decision tree trained by using the training corpus. The reason for the approximation in the equation is that we found that $b_e[i-1]$ and the information on the nearest words ($\psi[i]$ and $\psi[i+1]$) were the only important factors. The POS is used for the information about a word.

(c) Accent determination

The objective in this layer is to determine the accent sequence, $\mathbf{A} = (\alpha[0]\alpha[1]..\alpha[N_m-1])$, for the given PP, where $\alpha[i]$ has a value of `H` or `L`. A PP must have one of the accent types illustrated in the Fig. 3.5. Since the scope of this section is limited to the PP, we can use $\Psi = (\psi[0]\psi[1]..\psi[N_w-1])$ as the word sequence of the PP and $\mathbf{V} = (v[0]v[1]..v[N_m-1])$ as the sequence of the acoustic feature vectors without risk of confusion. The value of N_w is the number of words in the PP, and $N_m = \sum_{i=0}^{N_w-1} N_m[i]$ is the total number of morae in the PP, where $N_m[i]$ denotes the number of morae in the word $\psi[i]$. The objective can be restated as

$$\mathbf{A}_{max} = \operatorname*{\mathbf{argmax}}_{\mathbf{A}} P(\mathbf{A}|\Psi,\mathbf{V}) \tag{3.12}$$

$$= \operatorname*{\mathbf{argmax}}_{\mathbf{A}} \frac{P(\mathbf{V}|\Psi,\mathbf{A})P(\mathbf{A}|\Psi)}{P(\mathbf{V}|\Psi)} \tag{3.13}$$

$$= \operatorname*{\mathbf{argmax}}_{\mathbf{A}} P(\mathbf{V}|\Psi,\mathbf{A})P(\mathbf{A}|\Psi), \tag{3.14}$$

where $P(\mathbf{V}|\Psi)$ in Eq. (3.13) is ignored again. $P(\mathbf{V}|\Psi,\mathbf{A})$ and $P(\mathbf{A}|\Psi)$ are calculated by using the acoustic model and the linguistic model of this layer, respectively. We can obtain $\mathbf{A}_{max}$ by simply evaluating $P(\mathbf{V}|\Psi,\mathbf{A})P(\mathbf{A}|\Psi)$ for all of the cases of $\mathbf{A}$, since we assume only m possible sequences for $\mathbf{A}$.

Acoustic accent model We approximate $P(\mathbf{V}|\Psi, \mathbf{A})$ using the multiplication of $P(v[i]|\Psi, \mathbf{A})$s calculated by using a decision tree trained with the training corpus. A multivariate GMM is trained for each of the leaves of the tree. The components of the feature vector $v[i]$ are (1) the normalized logarithmic F0 at the beginning of the current mora, (2) the normalized logarithmic F0 change in the current mora, and (3) the logarithmic F0 gradient in the current mora (Fig. 3.8). We again approximate the possibility of the sequence by the product of the possibilities of the elements assuming the independence of the elements as follows.

$$P(\mathbf{V}|\Psi, \mathbf{A}) \simeq \prod_{i=0}^{N_m-1} P(v[i]|\Psi, \mathbf{A}) \tag{3.15}$$

$$\simeq \prod_{i=0}^{N_m-1} P(v[i]|\alpha[i-1], \alpha[i], N_m, i, (N_m - i)). \tag{3.16}$$

For Ψ and $\mathbf{A}$, the necessary input variables for the tree are the accents of the previous mora ($\alpha[i-1]$) and the current mora ($\alpha[i]$), the number of morae in the PP (N_m), and the distance to the beginning of the PP (i), and the distance to the end of the PP ($N_m - i$).

The normalized logarithmic F0 ($\widetilde{L0}$) is the logarithmic F0 normalized to fall in the range of $[0, 1]$ according to

$$\widetilde{L0} = \frac{L0 - L0_{min}}{L0_{max} - L0_{min}}, \tag{3.17}$$

where $L0_{min}$ and $L0_{max}$ are the minimum and maximum logarithmic F0s in the PP, respectively.

Linguistic accent model The linguistic probability $P(\mathbf{A}|\Psi)$ is obtained by employing a similar technique to the stochastic accent estimator [44]. $P(\mathbf{A}|\Psi)$, which is the probability of Ψ given one of the m possible accent sequences, is calculated by the following normalization of the joint probability of $\mathbf{A}$ and Ψ:

$$P(\mathbf{A}|\Psi) = \frac{P(\mathbf{A}, \Psi)}{\sum_{\forall \mathbf{A}} P(\mathbf{A}, \Psi)}, \tag{3.18}$$

where the summation of the denominator is used for all of the m possible accent sequences. The joint probability can be calculated as the product of the n-gram probabilities:

$$P(\mathbf{A}, \Psi) = \prod_{i=0}^{N_w-1} P(\alpha_s[i], \Psi[i]|\alpha_s[0], .., \alpha_s[i-1], \Psi[0], .., \Psi[i-1]), \tag{3.19}$$

where $\alpha_s[i]$, which is a part of $\mathbf{A}$, is the accent sequence of the i-th word. In other words, if the indices of the first and last mora of the word $\Psi[i]$ are j and k, respectively, $\alpha_s[i]$ is $(\alpha[j], ..., \alpha[k])$. The meaning of Eq. (3.19) can be roughly explained as approximating the probability of the accent sequence of the whole PP by the product of the probabilities of the constituent word accents. The n-gram probability $P(\alpha_s[i], \Psi[i]|\alpha_s[0], .., \alpha_s[i-1], \Psi[0], .., \Psi[i-1])$ is calculated from the n-gram frequency in the large speaker-independent text corpus. Note that a small constant probability is also given to unknown accent sequences. An example of the word sequence Ψ, the possible accent sequences A, and the calculated probabilities $P(\mathbf{A}, \Psi)$ and $P(\mathbf{A}|\Psi)$ is shown in Table 3.3.

Table 3.3: An example of the input (Ψ,$\mathbf{A}$) and the output ($P(\mathbf{A}|\Psi)$) of the linguistic accent model. $\mathbf{A} = (\mathtt{L}, \mathtt{H}, \mathtt{H}, \mathtt{L}, \mathtt{L})$ is the most likely linguistic sequence.

i	14	15	16	17		
Ψ	過ご Dosi su,go	せ Jodo se	な Jodo na	い Gobi i	$P(\mathbf{A}, \Psi)$	$P(\mathbf{A}\|\Psi)$
A	L,H	H	H	H	2.95e-14	0.315
	H,L	L	L	L	3.76e-32	4.02e-19
	L,H	L	L	L	3.38e-22	3.61e-09
	L,H	H	L	L	3.80e-14	0.406
	L,H	H	H	L	2.60e-14	0.278

Table 3.4: Statistics of the test corpus.

# of sentences	100
# of intonational phrases	377
# of prosodic phrases	686
# of words	1,813
# of morae	3,342
# of morae with an H accent	1,729 (51.7%)

3.2.3 Experimental results

We conducted experiments to evaluate the performance of the proposed method. In the experiments, we compared different combinations of the components of the method.

(a) Corpus

The speech corpus we used in the experiments is a reading of excerpts of the ATR 503-sentence text corpus [1]. We used different 100 sentences each for training the speaker-dependent models and for testing. Each sentence in the corpus was segmented into words and each word was manually annotated with its POS, its phoneme sequence, its accent sequence, and its PP boundaries. The sentences were separated into IPs at the positions of the manually labeled pauses with any length. For training the linguistic accent model, we used a speaker-independent text corpus of 28,351 sentences, including newspaper articles, TV news, telephone conversations, and other sources. The test sentences were excluded from the text corpus.

The speech data was recorded by an adult female using a laryngograph and a microphone. The F0 contours were obtained by smoothing the pitch mark periods obtained from the laryngograph. We used a Gaussian filter with $\sigma^2 = 0.00375$. The statistics of the test corpus are shown in Table 3.4. The mora length of the PPs in the corpus was between 1 and 13. Approximately 90% of the PPs contained 7 morae or less. Since every accent type for 1- to 7-mora PPs was found in the corpus, we considered the corpus had enough variety of accent types.

(b) Compared methods

We compared the following three combinations for PP boundary detection. The combinations are also shown in Table 3.5.

BA, BL and **BAL** The PP boundaries are detected by using either one or both of the acoustic and linguistic boundary models. **BAL** is our new proposed method for PP boundary detection.

We compared the following nine combinations for accent determination.

BN-TA, BN-TL and **BN-TAL** The mora accents were determined by using either one or both of the acoustic and linguistic accent models with a single-layer approach. In other words, there was no PP boundary detection. The accent sequences of the PPs were determined from all of the possible combinations. **BN-TL** is actually an approach to use the text processing component of the TTS system for accent determination, except that it is confining the output phonemes and POSs to the correct ones. If we assume that the output of the component is always one of the correct accentuations in the language, then the disagreement of this output and the test corpus with a speaker's actual speech reveals the arbitrary properties of the language or the speaking habits of the speaker.

BC-TA, BC-TL and **BC-TAL** For manually given correct PP boundaries, the mora accents were determined by using either one or both of the acoustic and linguistic accent models. The performance numbers of these methods can be viewed as the upper bounds of the accent determination algorithms alone, since the influences of the errors in PP boundary detection were eliminated from the numbers.

BAL-TA, BAL-TL and **BAL-TAL** Based on the PP boundaries detected by **BAL**, the mora accents were determined by using either one or both of the acoustic and linguistic accent models. **BAL-TAL** is our new proposed method.

(c) Results

The results of the PP boundary detection are shown in the upper part of Table 3.5. The accuracy is shown in precision, recall, and the F measure, which are calculated as follows: when the number of boundaries in the test corpus is N_a, the number of automatically detected boundaries is N_d, and the number of correctly detected boundaries is N_s, then the precision is N_s/N_d, the recall is N_s/N_a, and the F measure is $2\, \mathit{precision} \cdot \mathit{recall}/(\mathit{precision} + \mathit{recall})$. In another way of measuring performance, the accuracy of determining if each word boundary is a PP boundary or not was 93.7% for **BAL**. Note that the IP boundaries were ignored in the calculations of these values.

We can see that using both the acoustic and linguistic models (**BAL**) produced the best results. The poor precision of the acoustic model (**BA**) is an intrinsic problem of this model. This is because the acoustic features observed at a non-PP-boundary word boundary next to a real PP boundary are sometimes very similar to those observed at the real PP boundary, especially when the word sandwiched between these boundaries is very short. For example, postpositionals such as "wa", "ga", and "o" have only one mora.

The results of accent determination are shown in the lower part of Table 3.5. The three numbers for performance are the accuracy numbers measured by PPs, by words, or by morae as units, respectively. When calculating the PP accuracy, a PP is counted as correct only if the accents are correctly determined for all of the morae.

Table 3.5: The experimental results of the compared combinations. PPB stands for *Prosodic Phrase Boundary.* A and L stand for *acoustic*, and *linguistic*, respectively. While the performance numbers for PPB detection are precision, recall, and the F measure, those for accent determination are PP accuracy, word accuracy, and mora accuracy (%).

	PPB	Accent	Performance
BA	A	-	0.552/0.887/0.657
BL	L		0.693/0.896/0.781
BAL	A,L		0.821/0.906/0.862
BN-TA	Not used	A	25.7/48.6/62.4
BN-TL		L	62.0/79.1/85.5
BN-TAL		A,L	40.0/60.9/69.7
BC-TA	Correct	A	57.0/77.4/84.5
BC-TL		L	75.8/87.4/89.7
BC-TAL		A,L	84.1/91.8/94.6
BAL-TA	A,L	A	56.2/76.4/84.1
BAL-TL		L	68.9/83.4/87.8
BAL-TAL		A,L	77.6/88.4/92.7

Again, the combination of the acoustic and linguistic models showed the best performance (**BAL-TAL** > **BAL-TL** and **BAL-TA**, and **BC-TAL** > **BC-TL** and **BC-TA**). In addition, we can see the effectiveness of the proposed layered approach by comparing the result of the proposed method (**BAL-TAL**) and that of the non-layered approach (**BN-TAL**). Although the errors in PP boundary detection resulted in a 1.9-point decrease for accent determination accuracy (**BAL-TAL** < **BC-TAL**), the combined result is still over 90% and is better than the other approaches, the text-only processing (**BN-TL**) and the non-layered approach (**BN-TAL**).

(d) Discussion

In this section, we discussed an automatic prosody labeling method by combining acoustic and linguistic models, and speaker-dependent and speaker-independent models. The method showed 92.7% mora accuracy, which was higher than that of the previous work, based on a speaker-dependent training corpus containing only 100 sentences. We showed that the n-gram linguistic model that also uses the spelling of the words can serve as a rich information source about the possible correct accentuations in Japanese, while linguistic information had not been used for prosody labeling research for Japanese probably because word-level POS information was not sufficient in Japanese. The reason we reduced the required size of a speaker-dependent speech corpus using a large speaker-independent text corpus was that a text corpus is easier to gather than a speech corpus and a speaker-independent corpus is easier to gather than a speaker-dependent corpus. If we could measure the cost of gathering a corpus by price or time, the effectiveness of our method would be clearer.

To use both the speaker-dependent models and the speaker-independent models, the decomposition of the problem into two layers was inevitable. If we could simultaneously estimate the boundaries and the accents in some way, higher total accuracy might be achieved. The method proposed by Campbell [11] can detect the PP boundaries and the accent types simultaneously by searching for the prosodic label sequence that results in a synthetic voice matching the input speech best. Although the accuracy of the method was good, the method implicitly uses a linguistic knowledge embedded in the text processing module. Hence, the method can recognize only accent sequences in a candidate list generated by the text processing module. In addition, the method cannot be used for speakers whose F0 contour does not resemble the F0 contour of the synthetic voice.

It could be true that our method oversimplifies some aspects of the problem. Future work includes the following research items for more elaborate modeling. While our method takes the relationship among neighboring components into consideration for linguistic models, the acoustic models are based on naive Bayes models assuming mutual independency of the acoustic feature vectors. Furthermore, the acoustic boundary model ignores the effect of preceding prosodic phrase boundaries. It is possible that use of these currently ignored relationships improves the accuracy. Instead of the current simple handling of mora accents with `L` and `H`, modeling `H` morae in the accent type 0 and `H` morae in the other accent types differently may also improve the accuracy of the models. Enlarging the modeling unit from a mora to a prosodic phrase would have a similar effect. However, this would require a larger training corpus to model various types of prosodic phrases. Our handling of accent nuclei and phrase boundaries was also simple. Consideration of secondary accent nuclei and levels of phrase boundaries may contribute to more accurate models. In addition, such kind of more elaborate modeling would be useful especially for syntax analysis and emotion recognition. Soft decision of mora accents instead of hard classification of mora accents into `L` and `H` could be natural and useful for emotion recognition. By these research items, the accuracy of automatic prosody labeling will be further improved. The improved prosodic labels would make it easier to reproduce the speaker's speaking style more accurately with less manual efforts. By using more accurate modeling and speaker adaptation of the text processing module, recognition and reproduction of dialects such as Kansai accents may become possible.

3.3 Automatic Problematic Segment Detection

This step automatically detects problematic speech segments that should be removed from the segment DB.

3.3.1 Problematic segment detection based on likelihood metrics

The key features of the proposed method are:

- The method uses both acoustic metrics and duration-related metrics as clues to detect problematic segments.
- The acoustic metric of a segment indicates the acoustic reliability of the phonemic alignment. When a phonemic label is assigned to speech segments that are actually different phonemes, the acoustic likelihood should have a small value.
- The duration metric of a segment indicates properness of the duration of the segment. This works because unusually long or unusually short speech segments tend to accompany transcription errors or segmentation errors.

Although acoustic likelihood metrics are a straightforward measure for the reliability of the phonemic alignments, the existence of alignment errors and recognition errors indicates that use of acoustic likelihood metrics alone cannot solve the problem since these tools are actually based on acoustic likelihoods. On the other hand, as these tools do not use duration-related features in general, there is a good possibility that introduction of duration-related features improves the detection accuracy.

(a) Acoustic likelihood metrics

The acoustic likelihood metric $m_c[i]$ for the i-th segment is calculated referring to the acoustic model by the forced alignment mode of the ASR. The acoustic model is trained by the following steps:

1. A 13-dimensional cepstrum vector that consists of the energy of the frame and 12 cepstrums is calculated for each of GCIs of a speech.
2. By analyzing the neighboring cepstrums, dynamic features of the cepstrum vector are calculated.
3. A 39-dimensional vector is constructed by combining the cepstrum vector and the dynamic features.
4. A decision tree is built for each phoneme identification to predict the vector. The phonemic context of each phoneme is used as the input feature of the tree.
5. A multi-variate GMM is trained for each of the leaves to model the 39-dimensional vectors in the leaf.

The acoustic likelihood metric for a speech segment is calculated as follows:

1. The leaf is obtained by traversing the decision tree with the phonemic context of the segment.
2. For each of the GCIs in the segment, the 39-dimensional vector is calculated.
3. The acoustic logarithmic likelihood of the vector is calculated by referring to the GMM for the leaf.
4. The acoustic likelihood metric $m_c[i]$ is calculated as the mean of the acoustic logarithmic likelihoods for all the GCIs in the segments.

(b) Duration likelihood metrics

The value $m_d[i]$ of the duration likelihood metric for the i-th segment is calculated with the following steps:

1. Build a decision tree that predicts the duration of each phoneme [20]. The input features of the tree are the identity, the position, voicing of the phoneme, the POS of the word, the type of the sentence, etc.
2. Model the distribution of the phoneme durations in each leaf of the tree as Gaussian distributions.
3. Calculate the likelihood of the duration of each phoneme.
4. Remove the phoneme durations whose likelihood is below some threshold.
5. Model the distribution of the remaining phoneme durations again.

6. Calculate the logarithmic likelihood of the duration of each of the phonemes including the removed ones.
7. Take the mean of the likelihood in the neighborhood and let this average metric be $m_d[i]$.

In this way, it is possible to calculate the likelihood metrics of the durations without being affected by the irregular durations of the problematic segments.

(c) Viterbi search

After the acoustic likelihood metric $\mathbf{R} = (m_c[0], .., m_c[N_s - 1])$ and the duration likelihood metric $\mathbf{D} = (m_d[0], .., m_d[N_s - 1])$ are observed for all the segments in the speech, it is determined whether each of the segments is problematic or not. We define a variable $e_g[i]$ for i-th speech segment to indicate whether it is problematic ($e_g[i] = 0$) or not ($e_g[i] = 1$). Since problematic segments tend to occur in consecutive groups, we define the sequence $\mathbf{E}_g = (e_g[0], .., e_g[N_s - 1])$ of $e_g[i]$s and simultaneously calculate the values of $\mathbf{E}_g$ by considering the n-gram probability $P(e_g[i]|e_g[0], .., e_g[i-1])$, where N_s is the number of speech segments in the speech. A segment goodness sequence $\mathbf{E}_g = (e_g[0], .., e_g[N_s - 1])$ that maximizes the posterior probability in the following equation is searched for by using the Viterbi algorithm.

$$\begin{aligned}
\mathbf{E}_{g_{max}} &= \underset{\mathbf{E}_g}{\mathbf{argmax}}\, P(\mathbf{E}_g|\mathbf{R}, \mathbf{D}) && (3.20)\\
&= \underset{\mathbf{E}_g}{\mathbf{argmax}}\, P(\mathbf{R}, \mathbf{D}|\mathbf{E}_g)P(\mathbf{E}_g) && (3.21)\\
&\simeq \prod_{i=0}^{N_s-1} P(m_c[i], m_d[i]|e_g[i]) \prod_{i=0}^{N_s-1} P(e_g[i]|e_g[0], .., e_g[i-2]), && (3.22)
\end{aligned}$$

where $P(\mathbf{R}, \mathbf{D})$ is ignored in Eq. (3.21). Equation (3.22) approximates the possibility of the sequence by the product of the possibilities of the elements.

3.3.2 Experimental results

We make a training corpus by manually labeling the segment goodness $\mathbf{E}_g$ of the speech segments. Then $P(m_c[i], m_d[i]|e_g[i])$ is modeled by a multivariate GMM for each $e_g[i] = 0$ and $e_g[i] = 1$. Although the n-gram probability $P(e_g[i]|e_g[0], .., e_g[i-1])$ also can be trained by using the training corpus, we found that manually changed values resulted in better performance, so we used manually determined values of $P(e_g[i] = 0|e_g[i-1] = 1) = 0.0001$ and $P(e_g[i] = 1|e_g[i-1] = 0) = 0.15$.

In the experiments, we used 100 sentences each for training and testing. The test corpus includes 17,202 speech segments in total. There were 324 problematic segments that were found in 46 parts of the speech in the test corpus. The precision, recall, and the F measure of the proposed method were 0.298, 0.452, and 0.359. Although this was not good enough for some purposes, it was still better than the cases where either $\mathbf{R}$ or $\mathbf{D}$ was not used (0.288 and 0.331 respectively). When $P(e_g[i]|e_g[0], .., e_g[i-1])$ is not used, the F measure was 0.150.

(a) Discussion

Conventional duration-based approaches tend to improve the detection of problematic segments at the expense of precision. Unlike these approaches, since we think the precision is also important for large CTTS systems, we combined the acoustic likelihood metrics, $m_c[i]$, and duration likelihood metrics, $m_d[i]$, and achieved a relatively high F measure. However, the performance was still not very high. This was related to false positives. Sometimes the acoustic likelihood metrics or the duration likelihood metrics have erroneously high values when there are only a few substitutions of the recognized phonemes. Modeling alignment errors and recognition errors differently may have a good effect on the likelihood metrics. Explicit handling of alignment displacement amounts is also necessary for more precise modeling of problematic segments.

3.4 Concluding Remarks

In this chapter, we introduced a totally trainable text-to-speech (TTS) system, every component of which can be automatically built from human speech alone. We showed the procedure to incrementally collect the necessary linguistic and acoustic information from the speech by combining acoustic processing and linguistic processing. We described concrete means for all of the necessary sub-modules.

The accuracies of the sub-modules were examined by the experiments. By comparing the word error rates (WERs) of some transcription methods, it was shown that TTS voice generation using automatic speech recognition (ASR) without the transcription is at a practical level compared to the conventional TTS voice generation using transcriptions. The automatic prosody labeling algorithm achieved an F measure of 0.862 for prosodic phrase boundary detection by using the linguistic information (POS) with acoustic features, which was better than the results reported in previous research. For the accent determination problem of Japanese, which was a difficult task in which truly syllable-level judgment is required, we achieved 92.7% mora accuracy by using an n-gram linguistic model, including the spelling of the words. The required size of a speaker-dependent speech corpus was reduced to 100 sentences by using a speaker-independent text corpus. The overall quality of the synthetic voices was evaluated by the subjective listening tests. We can see from the results that the quality of the synthetic voices for in-domain text was assessed to be greater than 4.0 in the five-level subjective tests and approached the quality of the recorded human voices.

Although we were able to verify the validity of the framework, there remain some challenges for totally automatic generation of a new TTS voice. Preparation of a small training corpus for training the sub-modules was still necessary. Since accumulation of slight errors in the sub-modules results in non-ignorable errors in the generated TTS voice, we need to improve the accuracies of the sub-modules. The acoustic quality for out-of-domain text was assessed to be far worse than the acoustic quality for in-domain text. Further improvement of the algorithm is required to achieve higher acoustic quality especially for out-of-domain text. Although we have tested this approach only for Japanese, we believe our approach could also be effective for other languages including English. Since all the major components of the system are trainable, we can expect the system is easily applied to other languages or other Japanese accents.

Chapter 4

Robust Audio Watermarking Algorithm

In this chapter, we describe a robust audio watermarking algorithm that preserves the naturalness of the sound while making it possible to detect the watermark even after the sound quality is degraded. Since conventional time-domain spread spectrum watermarking algorithms require strict synchronization of the watermark signal, their robustness against attacks that displace the watermark signal was problematic. Therefore, we introduce a robust audio watermarking method that has advantages in both robustness and acoustic quality by modifying the magnitudes of the sound according to a two-dimensional pseudo-random array (PRA) defined in the time-frequency domain of the sound. In addition, the use of multiple stretched PRAs in the detection algorithm further improves the robustness against geometric distortions of the sound without requiring too much additional computational time. The communication capacity of the algorithm is also analyzed in the last part of the chapter. This chapter is related to the work published in [75, 74, 64, 67].

4.1 Audio Watermarking Algorithm Using a Two-Dimensional Pseudo-Random Array

In this section, we describe a multiple-bit audio watermarking method which is robust against geometric distortions of audio. For example, a 64-bit message can be detected in a 30-second music sample and survive random stretching, wow-and-flutter, and pitch shifting as well as MPEG compression, additive noise, echo, and digital-analog conversions. A psychoacoustic model calculates the amount of inaudible modification for watermark embedding and hence assures high acoustic quality. The detection algorithm does not need to refer to the original content.

The main ideas of the method can be briefly summarized as follows:

- The frequency-domain embedding algorithm uses a psychoacoustic model in a straightforward way. It modifies the magnitudes of the audio content by the amounts that the psychoacoustic model estimated human ears cannot distinguish.

- The embedding algorithm modifies the magnitudes of segmented areas in the time-frequency plane of the content, according to a two-dimensional pseudo-random array (PRA), while the detection algorithm correlates the magnitudes with the PRA. The two-dimensional array makes the watermark robust against cropping because, even when

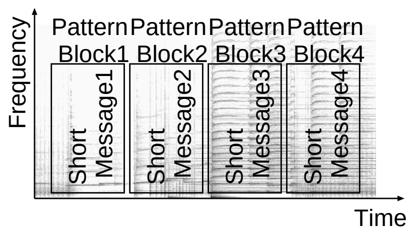

Figure 4.1: A pattern block has a two-dimensional area in the time-frequency plane and conveys a short message.

some portions of the content are heavily degraded, other portions of the content can match the PRA and contribute to watermark detection.

- The magnitude modification through overlap-and-add enables detection even from displaced detection windows. This is because magnitudes are less influenced than phases by fluctuations of the analysis windows caused by random cropping,

- Wider bandwidths at higher frequencies keep the correspondence of the embedded and detection PRA even for pitch-shifted content.

After the basic concepts for the method are described in Section 4.1.1, the algorithms for watermark embedding and detection are presented in Section 4.1.2 and Section 4.1.3, respectively. Theoretical and experimental analysis of the method is given in Section 4.1.4.

4.1.1 Basic concepts

Multiple-bit embedding This method can embed a multiple-bit message. For an instance, a 64-bit message is embedded in the robustness tests described in Section 4.1.4. The embedding algorithm should repeat the message and apply error-correcting/detection codings to it, and must add several bits to indicate the beginning of the message. The error-correcting/detection codings used in Section 4.1.4 can be found in [56]. After the codings, the multiple-bit message is divided into short messages, which are embedded separately.

Pattern blocks A short message is embedded in a *pattern block*, which is defined as a two-dimensional segmented area in the time-frequency plane of the content. The time-frequency plane is a two dimensional array constructed from the sequence of power spectrums calculated using short-term Discrete Fourier Transforms (DFTs). Figure 4.1 illustrates four consecutive pattern blocks in the time-frequency plane. The background image of the figure is a spectrogram of a music sample.

A pattern block is further divided into B_W and B_H tiles in rows and columns, respectively. Hence, the total number of tiles in a pattern block, B_A, is given by $B_H \times B_W$. We call B_W tiles in row a *subband*. A tile is the primitive for magnitude modification and contains several frequency components of four consecutive DFT frames. The B_A tiles in a pattern block share a synchronization signal and N_b bits (Fig. 4.2(a)). In the figure, B_H and B_W are 6 and 4, respectively. The synchronization signal is needed so the detection process can search for the beginning of a pattern block. It will be explained in Section 4.1.3 why this search is not computationally expensive. Hereafter, the subscript 'S' is used for the synchronization signal. One bit is encoded in W_B tiles, and W_S tiles are used for the synchronization signal.

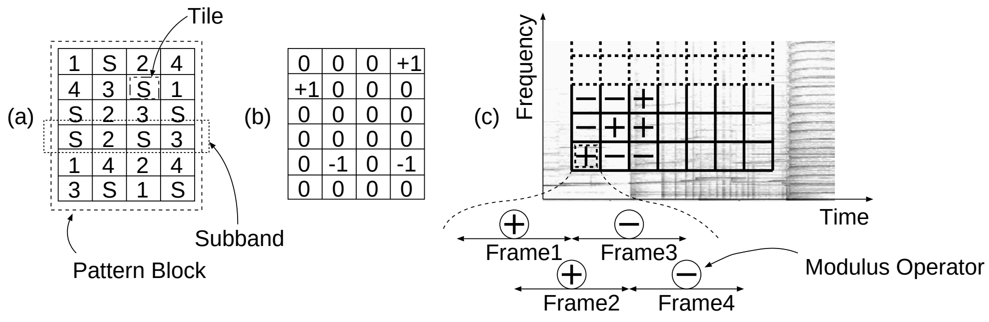

Figure 4.2: A pattern block consists of tiles and is shared by a synchronization signal and multiple bits. A tile contains four overlapping DFT frames. The tiles with S symbols are the tiles assigned for the synchronization signal.

Therefore we can write $B_A = W_B \times N_b + W_S$. The relationship of the frequency components, the subbands, and the tiles is shown in Fig. 4.10.

Pseudo-random arrays (PRA) A pseudo-random number corresponds to a tile. The embedding algorithm slightly modifies the magnitude of a tile according to the pseudo-random number assigned to the tile, and the detection algorithm correlates the magnitudes of tiles with the pseudo-random numbers. Therefore the position and value of the pseudo-random array (PRA) is a sort of secret key that must be shared by the embedder and the detector. We denote the two-dimensional pseudo-random array (PRA) by $\boldsymbol{\omega_F}$. A component, $\omega_F[t, b]$, of the array is the pseudo random number for the tile at (t, b) and has a value of $+1$ or -1. The set of the tile positions assigned for the j-th bit is $L[j]$, whose size is W_B. A tile position is denoted as (t, b), which means the tile is in the t-th column and in the b-th subband. The set of the tile positions assigned for the synchronization signal is L_s, whose size is W_S. Figure 4.2(a) illustrates a pattern block with four bits and a synchronization signal, and (b) is an example of $\boldsymbol{\omega_F}$ assigned for the fourth bit of (a).

Using a two-dimensional pseudo-random array, the method is robust against time fluctuation caused by duplicated or missing audio samples. This is because the loss in some portions of the array can be recovered by from other portions. The duration of a pattern block is an important factor for the robustness of the method. That will be shown in Section 4.1.4. Another advantage of the two-dimensional array is security. Varying the modification pattern of the magnitudes from frame to frame makes it difficult for crackers to analyze the secret pseudo-random array.

DFT frames and modulus operator A tile has four frames of Discrete Fourier Transform (DFT) each of which overlaps the adjacent frames by a half window (Fig. 4.2(c)). The relationship of the tiles and the frames are shown in Fig. 4.3. A more concrete example is shown in 4.4, where the length N of a frame is 1,024 samples. In the figure, the double-circled embedding frames are the beginnings of the tiles. The signs in the circles are examples of the signs used in a subband for each of the frames. The four frames in a tile are given a *modulus operator*, which is one element of a predefined sequence,

$$C_m[f] = \begin{cases} +1 & (f \leq 1 \pmod 4) \\ -1 & otherwise \end{cases} . \tag{4.1}$$

The f-th element of the sequence corresponds to the f-th frame of the four frames in a tile.

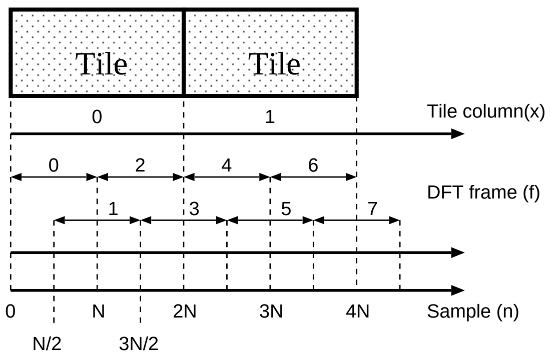

Figure 4.3: The relationship among tiles, frames and samples. A column of tiles corresponds to four frames. A frame consists of N consecutive PCM samples.

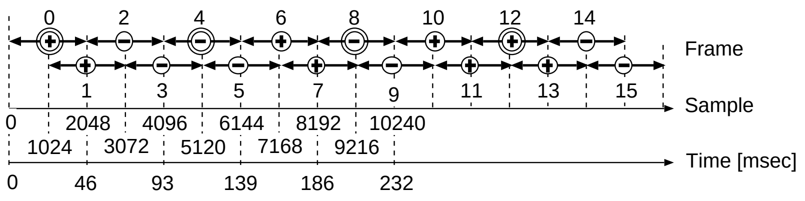

Figure 4.4: Frames used for embedding. The double-circled embedding frames are the beginnings of the tiles. The signs in the circles are examples of the signs used for each of the frames in a subband.

The embedding algorithm increases or decreases the magnitudes of a frame according to the sign of the pseudo-random value multiplied by the modulus operator. In other words, if the pseudo-random value assigned to a tile is positive, the embedding algorithm increases the magnitudes of frequencies of the former two frames in the tile and decreases those of the latter two frames. In this sense, we can say that we are using pattern blocks that have $4B_W$ DFT frames in total. Figure 4.5 is another illustration of the block shown in Fig. 4.2. In this figure, frames and the corresponding PRAs multiplied by the modulus operators are shown.

On the other hand, the detection algorithm calculates the difference of magnitudes of a frame and the next non-overlapping frame per tile and correlates the differences with the PRA. Hence, the pattern block regarding detection can be illustrated as Fig. 4.6.

The reason for using modulus operators is that since the magnitudes of adjacent frames have similar values in most cases, so this subtraction weakens the influence of the host signal on the detected watermark strength while it increases the effect of the watermark signal, because the opposite signs are embedded into the adjacent frames due to the modulus operator, C_{mf}. This idea is similar to Tsang's magnitude subtraction in video watermarking [78].

Subtraction pairs As described above, the detection algorithm calculates the difference of the magnitudes in a frame, f, and the next non-overlapping frame, $f + 2$. We call such a pair of two frames a *subtraction pair*. It can be said that the synchronization process selects a set of subtraction pairs by choosing the first frame of a pattern block that gives the maximum synchronization strength. When a set of subtraction pairs are selected, the magnitudes and watermark strengths calculated from the other set of subtraction pairs that overlap the first subtraction pairs by a half window are discarded (Fig. 4.7(a)). The signs in the circles are the

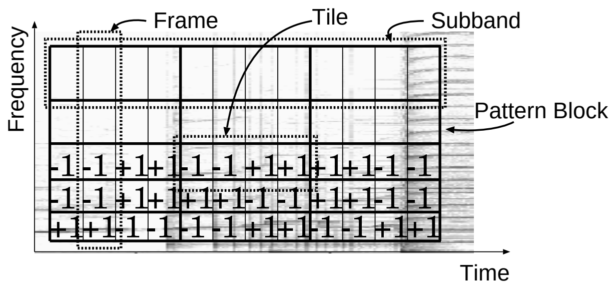

Figure 4.5: A pattern block drawn by expanding each column into four frames. The embedding algorithm modifies all of the magnitudes in the block.

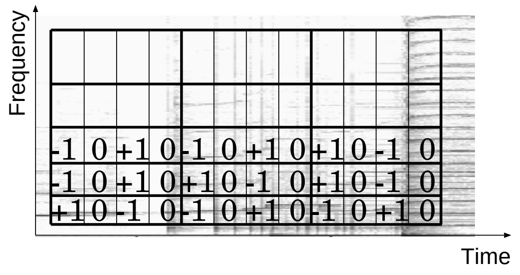

Figure 4.6: A pattern block for detection. The detection algorithm calculates the difference of magnitudes of a frame and the next non-overlapping frame per tile. It does not use the two other frames of the tile.

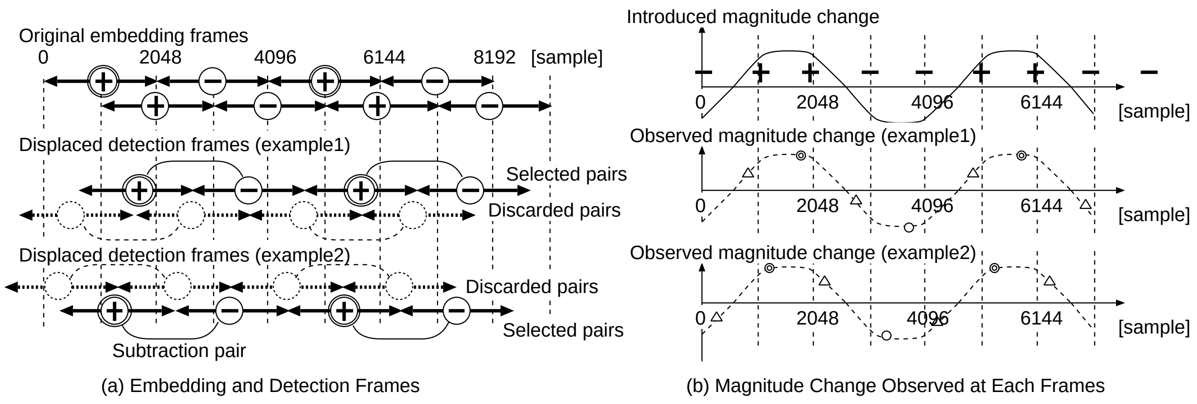
Original embedding frames
0
2048
4096
6144
8192
[sample]
Displaced detection frames (example1)
Selected pairs
Discarded pairs
Displaced detection frames (example2)
Discarded pairs
Selected pairs
Subtraction pair
(a) Embedding and Detection Frames
Introduced magnitude change
Observed magnitude change (example1)
Observed magnitude change (example2)
[sample]
(b) Magnitude Change Observed at Each Frames

Figure 4.7: On the left side of the figure are an illustration of original embedding frames and two examples of displaced detection frames. The illustrations on the right side are (1) magnitude changes introduced by watermark embedding and (2) magnitude changes that are expected to be observed at the displaced detection frames of the examples on the left side.

magnitude changes expected for those frames. The double-circled frames are the first frames of the selected pairs. By having two set of subtraction pairs, one of which is selected and the other is discarded, even when the detection frames are displaced from the original embedding frames, the magnitude modification introduced by the watermark embedding can be observed strongly from either set of pairs. Figure 4.7(b) is an illustration of the magnitude changes

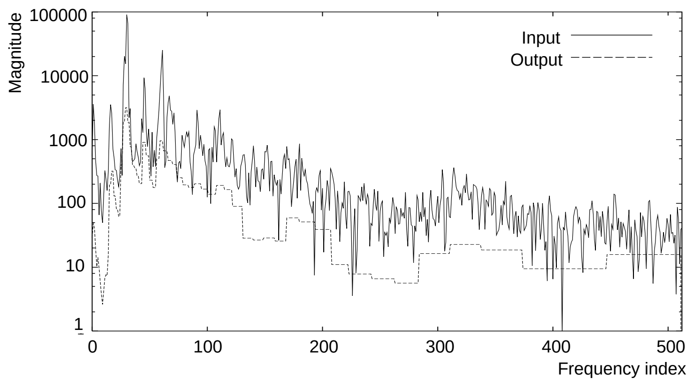

Figure 4.8: Input and output of the psychoacoustic model. The horizontal axis is the index of frequency components. The vertical axis is magnitudes of frequency components.

observed for the detection frames illustrated in Fig. 4.7(a). The solid line and the dashed lines are the magnitude changes made from the original embedding frames drawn in Fig. 4.7(a). The circles on the lines are the magnitude changes expected for the detection frames in the selected pairs. The triangles are the magnitude changes observed for the frames in the other set of pairs. Those will be discarded because the magnitude changes are less prominent in those frames.

Psychoacoustic model The embedding algorithm uses a psychoacoustic model for each frame to determine the magnitudes of the watermark signal. Among auditory masking effects [84], we use only simultaneous frequency masking effects for the model. Simultaneous frequency masking effects are the effects that a stronger masker frequency component makes a weaker frequency component whose frequency is close to the frequency of the masker inaudible. By using the psychoacoustic model, the masking effect at each frequency can be calculated as summation of the individual masking effects by other frequencies to the frequency. The obtained masking effects can be used as a reference of the magnitude change that human ears cannot notice. An example of the input and output of the psychoacoustic model used in the tests in Section 4.1.4 is shown in Fig. 4.8.

Although there are *threshold in quiet* effects and temporal masking effects for human auditory systems besides frequency masking effects, the use of these effects is not effective in case of audio watermarking. *threshold in quiet* effects are the effects that human ears cannot notice sounds below a threshold even when there is no masking sound. The value of the threshold is dependent on the frequency. The reason we cannot use these effects for audio watermarking is the following: If the embedder utilizes *threshold in quiet* and embeds a small watermark signal under the threshold in a silent region, and then if the listener increases the volume of the amplifier or the headphone, the small sound is no longer below the threshold and becomes audible. As for temporal masking effects, though utilization of these effects may lead to accurate calculation of inaudible magnitudes, the effects are less than those of simultaneous masking. However, it is important to exceptionally treat attacks of sound where the energy of the host signal rapidly increase. This is because pre-echo effects tend to occur in these regions as both Fourier transforms and psychoacoustic model cannot perform accurate analysis in these regions. Therefore, it is preferable to skip embedding watermark in these regions by detecting attacks.

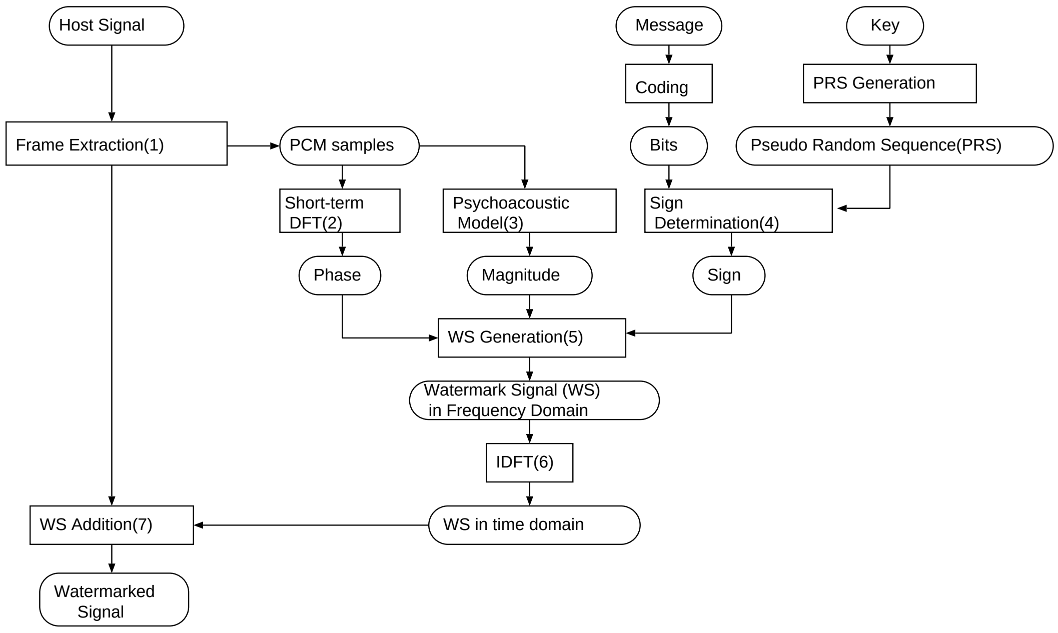

Figure 4.9: The process flow of the watermark embedding algorithm. The numbers in the parentheses correspond to the step numbers in the description.

4.1.2 Embedding algorithm

The embedding algorithm calculates a watermark signal in the frequency domain, converts it to the time domain using an inverse DFT (IDFT) and adds it into the host signal (Fig. 4.9). The magnitudes of the watermark signal are calculated using the psychoacoustic model.

Whether magnitudes of the host signal are increased or decreased is decided by a pseudo-random array and the bits to be embedded. The embedding algorithm uses the same phases as the phases of the host signal for constructing the watermark signal to avoid introducing unnecessary phase changes.

Step 1 Overlapping frames

This step extracts DFT frames. A frame consists of N consecutive PCM samples and overlaps the adjacent frames by a half window. We denote the N PCM samples for the f-th frame as $x[f,n]$ $(0 \leq n < N)$.

Step 2 Windowing DFT

The complex frequency components $c_c[f,k]$ $(0 \leq k < N/2)$ are obtained by the DFT analysis of the PCM samples $x[f,n]$ multiplied by a windowing function $f_{win}(n)$. $c_c[f,k]$ is the k-th complex frequency component of the f-th frame.

$$c_c[f,k] \quad = \quad \mathrm{DFT}\left[x[f,n]f_{win}(n)\right]. \tag{4.2}$$

The amplitude $c_a[f,k]$, the phase $c_\phi[f,k]$, the real component $c_r[f,k]$, the imaginary component $c_i[f,k]$ of the frequency component are calculated by the following equations.

$$c_a[f,k] \quad = \quad |c_c[f,k]|\,, \tag{4.3}$$
$$c_\phi[f,k] \quad = \quad \arg\left(c_c[f,k]\right), \tag{4.4}$$
$$c_r[f,k] \quad = \quad Re\left[c_c[f,k]\right], \tag{4.5}$$
$$c_i[f,k] \quad = \quad Im\left[c_c[f,k]\right]. \tag{4.6}$$

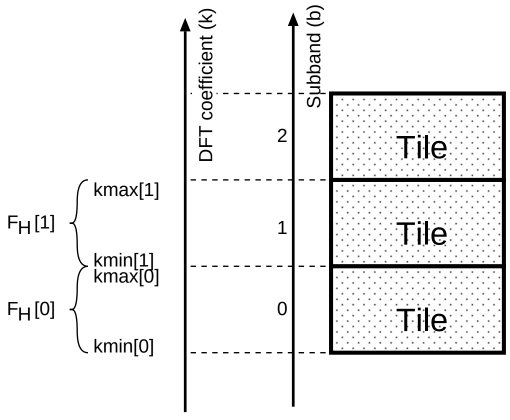

Figure 4.10: The relationship among tiles, subbands, and frequency components. A subband is a row of tiles. The b-th subband consists of $F_H[b]$ frequency components.

Step 3 Calculation of inaudible modification amount

Then the algorithm calculates the inaudible level of the magnitude modification by using a psychoacoustic model based on the complex spectrum. We indicate this amount of the k-th frequency of the f-th frame in a pattern block by $a_p[f, k]$. We use this amount for the magnitude in the k-th frequency component of the watermark signal.

Step 4 Watermark sign determination

This step calculates the signs that determine whether to increase or decrease the magnitudes of the host signal in each of the tiles. $s_{tile}[t, b]$ is the sign for the b-th subband in the t-th column of the pattern block. The tile includes four DFT frames whose frame index f is in the range of $4t \leq f < 4(t + 1)$. Let the tile is assigned for the j-th bit. That is, $(t, b) \in L_j$. If the sign $s_{bin}[f, k]$ calculated based on these values is positive, the k-th amplitude frequency component of the f-th frame is increased. Otherwise, the amplitude is decreased.

$$\begin{aligned} s_{bin}[f,k] &= C_m[f]s_{tile}[t,b] && (4.7)\\ &= C_m[f](2m[j]-1)\omega_F[t,b], && (4.8) \end{aligned}$$

where $C_m[f]$ is the modulus operator.

In the tiles where the calculated sign, $s_{tile}[t, b]$, is positive, the phase of the HS, $c_\phi[f, k]$, is used for the phase in the f-th frequency bin of the WS, while we assume the k-th frequency is in the b-th subband. In the tiles with a negative sign, the opposite phase $-c_\phi[f, k]$ is used.

Step 5 Watermark signal generation

This step calculates the complex frequency components of the watermark signal. We use the phases c_ϕ of the host signal as the phases of the watermark signal. We use the amplitudes a_p of the output of the psychoacoustic model as the absolute amplitudes of the watermark signal. While the watermark signal is generated to increase the amplitude of the host signal in the tiles where the sign $s_{tile}[t, b]$ is positive. Where $s_{tile}[t, b]$ is negative, the phase of the watermark signal is exactly the opposite of that of the host signal in the tile. That results in decreasing the magnitude of the host signal in the tile(Fig. 4.11).

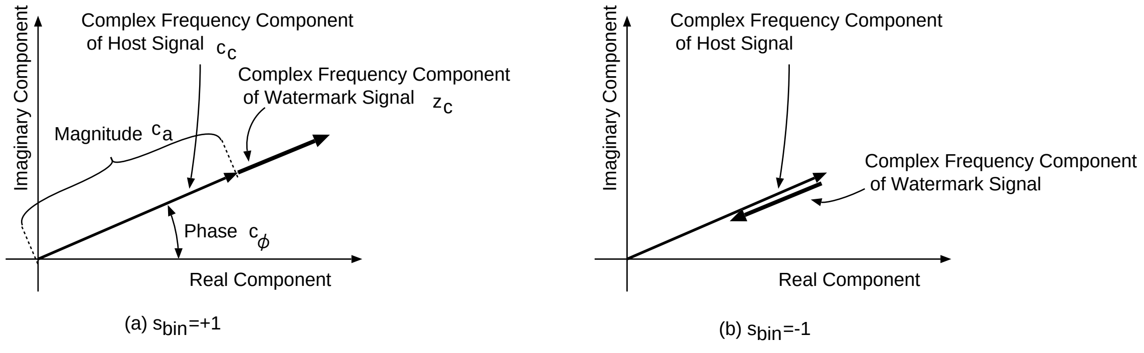
Imaginary Component
Complex Frequency Component of Host Signal c_c
Complex Frequency Component of Watermark Signal z_c
Magnitude c_a
Phase c_ϕ
Real Component
(a) s_{bin}=+1
Imaginary Component
Complex Frequency Component of Host Signal
Complex Frequency Component of Watermark Signal
Real Component
(b) s_{bin}=-1

Figure 4.11: The watermark signal in the frequency domain. The watermark signal modifies the magnitude of the host signal in the frequency domain.

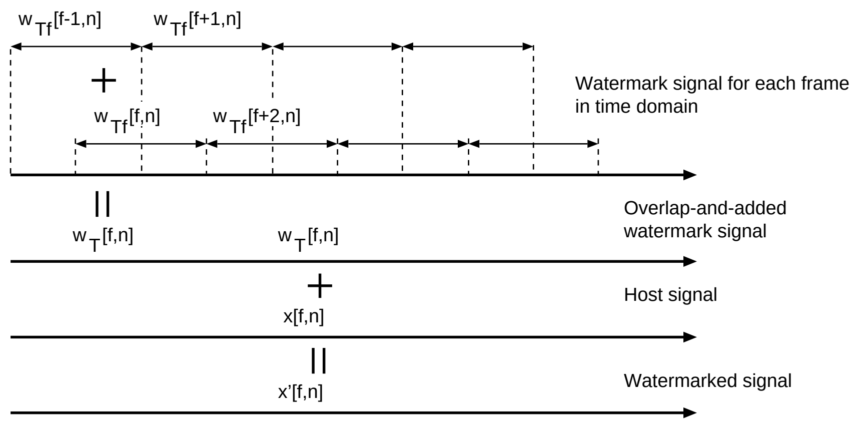

Figure 4.12: Watermark signal addition. The watermark signal is added to the host signal in the time domain. Two adjacent frames are overlapped.

$z_c[f,k]$ is the k-th complex frequency component of the WS in the f-th frame (Fig. 4.11).

$$z_c[f,k] = s_{bin}[f,k]a_p[f,k]\exp\left(c_\phi[f,k]\sqrt{-1}\right), \tag{4.9}$$

where the frequency index k is included in the b-th subband ($\mathrm{kmin}[b] \leq k \leq \mathrm{kmax}[b]$).

Step 6 IDFT, windowing and overlap-and-add (OLA)

The watermark signal in the time domain is obtained by transforming these values using IDFTs. To avoid generating clicking sounds at the borders of adjacent IDFT frames, the watermark signal is multiplied by a windowing function and overlapped with the adjacent frames after each IDFT. The first half ($0 \leq n < \frac{N}{2}$) of a frame is overlapped with the latter half of the previous frame, while the second half ($\frac{N}{2} \leq n < N$) is overlapped with the first half of the next frame (Fig. 4.12).

$$\omega_{Tf}[f,n] = f_{win}(n)\mathrm{IDFT}\left[z_c[f,k],\right], \tag{4.10}$$

$$\omega_T[f,n] = \omega_{Tf}\left[f-1, n+\frac{N}{2}\right] + \omega_{Tf}[f,n] \quad \left(0 \leq n < \frac{N}{2}\right), \tag{4.11}$$

$$\omega_T[f,n] = \omega_{Tf}[f,n] + \omega_{Tf}\left[f+1, n-\frac{N}{2}\right] \quad \left(\frac{N}{2} \leq n < N\right). \tag{4.12}$$

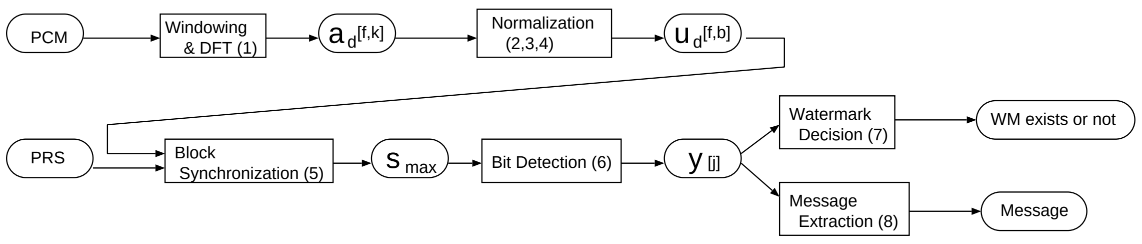

Figure 4.13: The process flow of the watermark detection algorithm. The numbers in the parentheses correspond to the step numbers in the description.

Step 7 Watermark signal addition
Finally, the watermarked PCM samples are obtained as the summation of the host signal and the watermark signal in the time domain.

$$x'[f,n] = x[f,n] + \omega_T[f,n]. \tag{4.13}$$

4.1.3 Detection algorithm

The detection algorithm calculates the magnitudes for all tiles of the content and correlates them with the pseudo-random array (PRA) by applying the following steps (See Fig. 4.13)):

Step 1 Windowing DFT
The magnitude $c_a[f,k]$ of the k-th frequency in the f-th frame of a pattern block of the content is calculated by the DFT analysis of a frame of the host signal. A frame consists of N consecutive PCM samples and overlaps the adjacent frames by a half window. The samples should be multiplied with a windowing function such as a sine window before the DFT.

Step 2 Normalization
The magnitudes are then normalized by the mean of the magnitudes in the frame so that contributions of all frames to the watermark strength are equal. A normalized logarithmic magnitude is

$$\tilde{c_a}[f,k] = \log\left(\frac{c_a[f,k]}{\frac{1}{N/2}\sum_{k=0}^{N/2-1} c_a[f,k]}\right). \tag{4.14}$$

Step 3 Magnitude of tile
The magnitude of a tile located at the b-th subband of the f-th frame in the block is calculated as

$$u[f,b] = \frac{\sum_{k=\mathrm{kmin}[b]}^{\mathrm{kmax}[b]} \tilde{c_a}[f,k]}{F_H[b]}, \tag{4.15}$$

where kmin$[b]$ and kmax$[b]$ are the lowest and highest frequency indices in the b-th subband, respectively. $F_H[b] = \mathrm{kmax}[b] - \mathrm{kmin}[b] + 1$ is the number of frequency components in the tile.

Step 4 Tile value

The difference between the magnitudes of a tile and the next non-overlapping frame is taken as

$$u_d[f,b] = u[f,b] - u[f+2,b]. \tag{4.16}$$

This subtraction increases of the effect of the watermark signal taking advantage of the modulus operator.

Step 5 Pattern block synchronization

Because there is a possibility that the content has been trimmed before detection, the start of the content is not necessarily the beginning of a pattern block. Therefore, the beginning of a pattern block has to be located. We call this procedure *pattern block synchronization.*

Note that the minimum step of the search is a frame, which is much larger than a PCM sample. If a search is required for all samples, it would be very computationally expensive. However, pattern block synchronization requires only $4B_W$ calculations of synchronization strength, and additional DFTs are unnecessary because the search is after the DFTs.

First, a *synchronization strength*, $S[s]$, is calculated for each frame based on the assumption that the s-th frame is the beginning of the pattern block. The frame that gives the maximum synchronization strength is accepted as the beginning of the pattern block. The $S[s]$ is calculated as the normalized correlation of the PRA and the magnitude differences:

$$S[s] = \frac{\sum\limits_{\forall(t,b)\in L_s} \omega_F[t,b](u_d[4t-s,b] - \overline{u_d}[s])}{\sqrt{\sum\limits_{\forall(t,b)\in L_s} \{\omega_F[t,b](u_d[4t-s,b] - \overline{u_d}[s])\}^2}}, \tag{4.17}$$

where

$$\overline{u_d}[s] = \frac{1}{W_S} \sum_{\forall(t,b)\in L_s} u_d[4t-s,b], \tag{4.18}$$

and $\omega_F[t,b]$ is the pseudo-random number corresponding to the tile at b-th subband in the t-th column of the pattern block. If the content is unwatermarked, the expected value of $\omega_F[t,b](u_d[4t-s] - \overline{u_d}[s])$ is zero and the denominator of Eq. (4.17) should be the sample standard deviation serving as an approximation of the standard deviation. Hence, due to the Central Limit Theorem, the distribution of Eq. (4.17) can be approximated by a standard Gaussian distribution. That is true, even though the host signal is not Gaussian noise, because the usage of the pseudo random number $\omega_F[t,b]$ and the normalization make $\omega_F[t,b](u_d[4t-s,b] - \overline{u_d}[s])$ independently and identically distributed (i.i.d.).

Synchronization strengths are calculated with s from 0 up to $4B_W - 1$, because a pattern block has B_W tiles in row and a tile contains 4 frames. The position s maximizing the synchronization strength is chosen as

$$s_{max} = \mathop{\mathbf{argmax}}_{s=0}^{4B_W-1} S[s]. \tag{4.19}$$

Assuming that several consecutive pattern blocks have synchronization positions that are separated by the same number of frames, then the successful synchronization rate can be improved. This *linear assumption method* searches for synchronization positions for consecutive N_{LA} pattern blocks at the same time by searching the number of frames in a pattern block, d_l, and the synchronization position for the first pattern block, s_{max}. The *multiple pattern block synchronization strength* is given by

$$\overline{S}[d_l, s] = \frac{1}{N_{LA}} \sum_{q=0}^{N_{LA}-1} S[q \times d_l + s]. \tag{4.20}$$

If the pitch of the content is known not to have been shifted, d must be within $4B_W$, and a search for d_l is not necessary. Similarly, if the content has not been trimmed, s_{max} must be 0. The $d_{l_{max}}$ and s_{max} satisfying the following equation:

$$(d_{l_{max}}, s_{max}) = \mathop{\mathbf{argmax}}_{d_l=d_{l_{lower}}}^{d_{l_{upper}}} \mathop{\mathbf{argmax}}_{s=0}^{4B_W-1} \overline{S}[d_l, s], \tag{4.21}$$

give the synchronization positions for the blocks, where $d_{l_{lower}}$ is the shortest possible length of a block, and $d_{l_{upper}}$ the longest possible. The resulting synchronization position for the q-th block is $q \times d_{l_{max}} + s_{max}$.

In a case where the true synchronization positions for consecutive pattern blocks are shifted, choosing a local maximum value of S_s within a few neighboring frames improves the reliability of synchronization. We call this procedure *local adjustment.*

Step 6 Bit detection
Bit strengths, $y[j]$, are calculated at the obtained synchronization position.

$$y[j] = \frac{\sum_{\forall(t,b)\in L[j]} \omega_F[t,b](u_d[4t - s_{max}, b] - \overline{u_d}[s_{max}])}{\sqrt{\sum_{\forall(t,b)\in L[j]} \{\omega_F[t,b](u_d[4t - s_{max}, b] - \overline{u_d}[s_{max}])\}^2}}. \tag{4.22}$$

$\omega_F[t, b]$ is the pseudo-random number for the tile at (t, b). $L[j]$ is the set of tile positions assigned for the j-th tile. The sign of $y[j]$ indicates the value of the j-th bit in the pattern block.

$$r_j = \begin{cases} 1 & (y[j] \geq 0) \\ 0 & (y[j] < 0) \end{cases}. \tag{4.23}$$

Note that the distribution of bit strengths, $y[j]$, for unwatermarked content can be also approximated by a standard Gaussian distribution.

Step 7 Watermark decision
The decision on whether the content has been watermarked or not is made at this stage using $y[j]$. The null (H_0) and the alternative (H_1) hypothesis are that there is no watermark and that there is a watermark, respectively. In our case, since both the watermark signal and the host signal are unknown, it is difficult to calculate the likelihood ratio. Accordingly, we have chosen a hypothesis test described below as a practical method that makes it possible to set a constant false alarm rate. Under H_0, because $y[j]$ asymptotically follows the standard Gaussian distribution, the sum of the squares of $y[j]$, $y_a = \sum_{j=1}^{N_b} y[j]^2$, asymptotically follows a central chi-square distribution with N_b degrees of freedom, where N_b is the number of bits detected within a predefined length of the content. On the other hand, under H_1, the distribution of y_a becomes a non-central chi-square distribution and the expected value of y_a should be a large

value regardless of the actual watermark signal embedded in the content. Therefore, the decision can be made by comparing y_a with a predetermined threshold T_{wm}.

$$if \left(\sum_{j=0}^{N_b-1} y[j]^2 \geq T_{wm} \right) \textit{ then the content is watermarked.} \tag{4.24}$$

The advantages of this test are (a) flexibility for setting a constant false alarm rate, (b) applicability to an unknown watermark signal, and (c) good performance against non-Gaussian noise.

Furthermore, when the number of bits that is weaker than a predetermined threshold, T_{wk}, exceeds a certain number, T_{wb}, the detector should not output a message:

$$if \ \left(\| \{ \, j \mid |y[j]| < T_{wk} \} \| > T_{wb} \right) \textit{ then no output,} \tag{4.25}$$

where $\| X \|$ is the number of the components in the set X. This is because it is likely that there are too many bit errors to recover using error-correcting codes. The thresholds should be determined considering the capability of the error-correcting code.

Step 8 Reconstruction of the multiple-bit message
Finally, the multiple-bit message is reconstructed from the detected bits. Error correction and detection should also be performed at this stage.

4.1.4 Theoretical and experimental analysis

Theoretical and experimental analysis of the robustness of the method is shown using an experimental system. We also discuss the crucial parameters for robustness.

(a) Parameter design

We implemented the method in a software system that can embed and detect a 64-bit message in 30-second pieces of music. The message is encoded in 254 bits by adding 7 Cyclic Redundancy Check (CRC) parity bits, using a Bose-Chaudhuri-Hocquenghem (BCH(127, 71, 15)) code, and repeated twice. Each pattern block has 4 bits embedded, and the block has 30 columns and 9 rows of tiles. The bandwidths of the 30 frequency subbands are described in Section (c). 150 tiles out of the 270 tiles are assigned for the synchronization signal, 30 tiles for a bit. For the pattern block synchronization, 7 consecutive blocks are used for the linear assumption method. For the local adjustment, the four neighboring frames are evaluated. The length of a DFT frame is 1,024 samples. A sine window is used for windowing the DFT frames. The thresholds for deciding whether the content is watermarked or not are set so that the false alarm error ratio is under 10^{-6}.

Three pieces of music were used for the analysis: a violin sonata by Bach, a symphony by Debussy, and a female jazz vocalist with strings and a piano. All signals are sampled at a frequency of 44.1 kHz, and each piece is 100 seconds long. The resulting Signal-to-Noise ratio from watermark embedding was 35.0 dB on average. When JASRAC, CISAC, and BIEM conducted evaluation tests of audio watermarking technologies in STEP2000 [33] and STEP2001 [34], we submitted embedders that embed 74 bits within 30 seconds using psychoacoustic models almost identical to the psychoacoustic model used for this section's experiment. In the result, they certified the acoustic quality of the watermarked content through ABX tests by golden ears. The only major difference between the models used for STEP and the model used for this section is that, in the STEP version attacks of the sound were automatically detected using another algorithm and were left unwatermarked for acoustic quality.

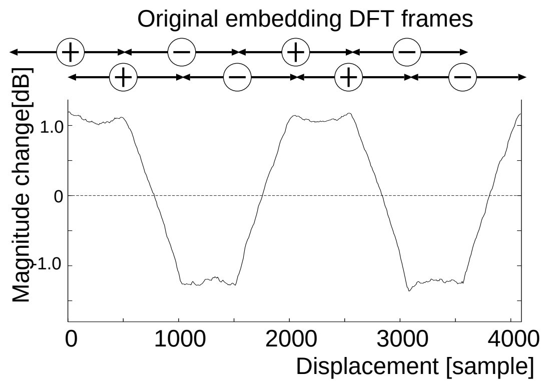

Figure 4.14: Magnitude change observed at displaced DFT frames.

(b) Effect of magnitude modification

One of the important characteristics of the method is modifying magnitudes that are independent of phases. Because magnitudes are less influenced than phases by displacement of the analysis windows, the watermark can be detected even after cropping. Although the shift-invariance characteristics of the magnitudes of the DFT are well-known, we show the practical effectiveness of the effect for audio watermarking in this section.

Figure 4.14 shows an experimental result of the magnitude modification by watermark embedding. The abscissa is the number of samples by which the observing DFT window is displaced, and the ordinate is the mean of the magnitude change before and after the watermark embedding. The arrows and signs above the graph indicate the original DFT windows used for watermark embedding. It can be seen that the modified magnitudes can be observed even at intermediate DFT frames between adjacent original embedding frames to which the same sign is embedded by the modulus operators. When the detection frames are located between the adjacent original embedding frames to which the different signs are embedded, the modified magnitudes cannot be effectively observed. However, in that case, the other sequence of frames which are overlapping with the ineffective frames by a half window and located between the original embedding frames to which the same sign is embedded , will give a strong detection strength (See Fig. 4.7).

Figure 4.15 shows the mean of the detected bit strength[1], $y[j]$, for the displacement from 0 up to 512 samples, which is the interval between two consecutive DFT frames. The mean drops slightly but remains sufficiently high up to 512 samples of displacement. Note that the next frame will be selected for more than 255 samples of displacement by the pattern block synchronization process. This data is the reason the detection algorithm does not need a sample-by-sample exhaustive search for the original DFT frames used for embedding.

(c) Bandwidth

The design of the pattern blocks is crucial for robustness. When the content is transformed in some way, the shapes of the blocks are frequently changed. When the pitch of the content is shifted upward, the duration of a block becomes shorter and the frequencies of the block become higher. When wow-and-flutter affects the content, the duration and the height of the blocks becomes different. In these cases, the detector cannot synchronize the pseudo-random array to the blocks that were embedded in the content, and the detected watermark

[1]To obtain a meaningful mean, 1 is embedded for all watermark bits.

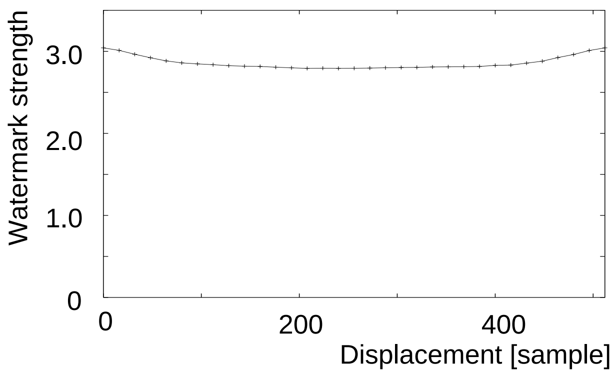

Figure 4.15: Averaged watermark strength detected from displaced DFT frames. The worst displacement is 256-sample which is the half of the interval of adjacent frames.

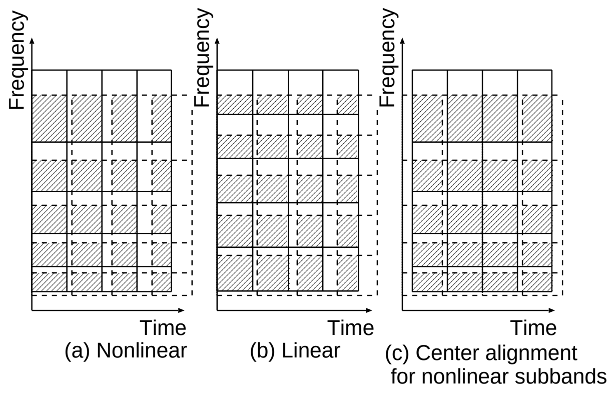

Figure 4.16: Design of pattern blocks. The broken and solid squares illustrate the shapes of the pattern blocks before and after pitch shifting, respectively. The hatched area maintains the correspondence of the embedded PRA and the detection PRA.

strength consequently decreases. In order for detection succeed in spite of these sorts of signal transformations, pattern blocks must have a shape that maintains the correspondence between the embedded watermark signal and the pseudo-random array.

The first design parameter for a pattern block is the bandwidth for the tiles. When the pitch of the content is shifted by a rate r_p, the lowest frequency, $\mathrm{kmin}[b]$, and the highest frequency, $\mathrm{kmax}[b]$, of the b-th subband become $r_p\mathrm{kmin}[b]$ and $r_p\mathrm{kmax}[b]$, respectively, and hence frequency components over $\mathrm{kmax}[b]$ no longer contribute to the subband. Therefore the contribution of the subband becomes $\langle \mathrm{kmax}[b] - r_p\mathrm{kmin}[b]\rangle / F_H[b]$ times the original contribution, where $\langle x \rangle$ is 0 if x is smaller than 0, otherwise x. With $r_b[b] = \frac{\mathrm{kmax}[b]}{\mathrm{kmin}[b]}$, the degradation rate can be expressed by

$$\frac{\langle \mathrm{kmax}[b] - r_p\mathrm{kmin}[b]\rangle}{\mathrm{kmax}[b] - \mathrm{kmin}[b]} = \frac{\langle r_b[b] - r_p\rangle}{r_b[b] - 1}, \tag{4.26}$$

which is independent of $\mathrm{kmin}[b]$. For this reason, by using the same r_b value for all $r_b[b]$, the contributions of all subbands degrades at the same rate, which is better than allowing some subbands to degrade more rapidly than other strong subbands. This idea is basically same as constant spacing in the log frequency domain, but that leads to bands that are too narrow at

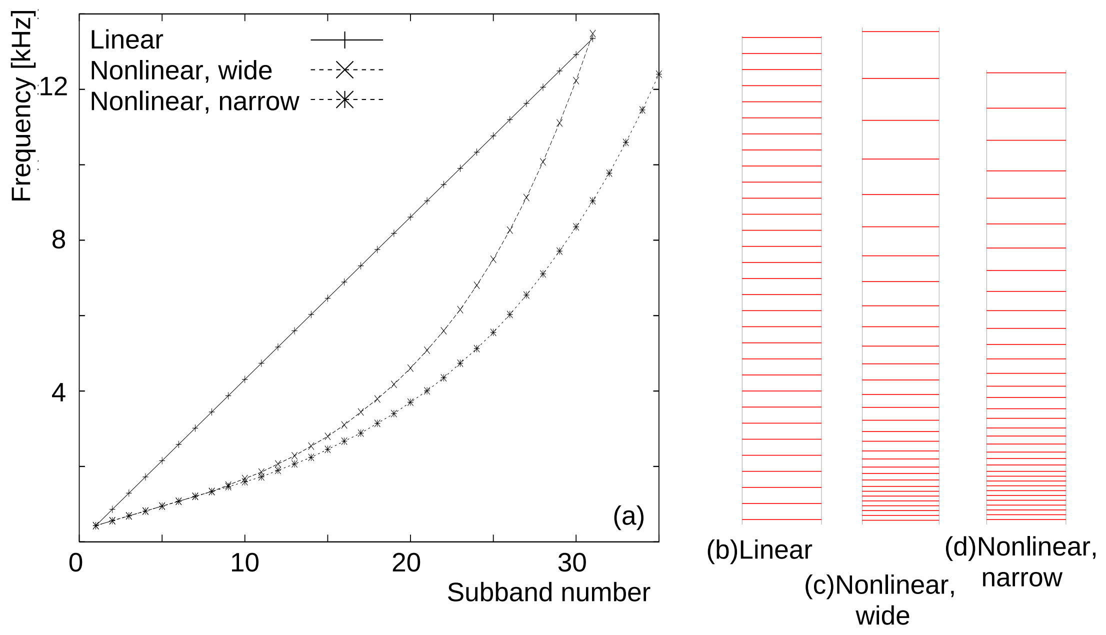

Figure 4.17: Three types of subbands. While every subband in (b) has the same width, (c) and (d) has wider subbands in higher frequency.

lower frequencies and bands that are too wide at higher frequencies and results in too few bands in total. Hence, we do not simplify the bandwidth design by introducing the log frequency domain. The dotted lines of Fig. 4.16(a) illustrates tiles with a common r_b value, and the solid line is the shape of the tiles after the pitch of the content has been shifted. The hatched area of the figure is the area maintaining the correspondence and hence still contributing to the detected watermark strength. On the other hand, (b) illustrates another pattern block where every subband has the same bandwidth. The hatched areas at high frequencies are smaller than at low frequencies.

The detected watermark strengths, $y[j]$, are proportional to the hatched areas, which decreases as the pitch-shifting rate, r_p, increases. We call the decreasing rate of the hatched area the *correspondence* rate. The correspondence rate in the case of Fig. 4.16(a) or (b) can be estimated by the following simple geometrical calculation:

$$r_d(r_p) = \sum_{b=1}^{B_H} \sum_{t=1}^{B_W} \frac{\langle \mathrm{kmax}[b] - r_p \mathrm{kmin}[b] \rangle}{\mathrm{kmax}[b] - \mathrm{kmin}[b]} \left\langle \frac{t}{r_p} - (t-1) \right\rangle . \tag{4.27}$$

The reason for the proportionality is that the numerator of Eq. (4.22) is proportional to the maintained correspondence between $\omega_F[t,b]$ and the pseudo-random array embedded in the $u_d[f,b]$s while the denominator of $y[j]$ is basically independent of the correspondence.

Note that the actual correspondence to be found by the pattern block synchronization process is expected to be Fig. 4.16(c) instead of Fig. 4.16(b). This is because Fig. 4.16(c), where the centers of the original and shifted pattern blocks match, maximizes the hatched area and the synchronization process finds the position that maximizes this area. The correspondence rate in that case is estimated by

$$r_d(r_p) = \sum_{b=1}^{B_H} 2 \sum_{t=1}^{B_W/2} \frac{\langle \mathrm{kmax}[b] - r_p[b] \rangle}{\mathrm{kmax}[b] - \mathrm{kmin}[b]} \left\langle \frac{t}{r_p} - (t-1) \right\rangle . \tag{4.28}$$

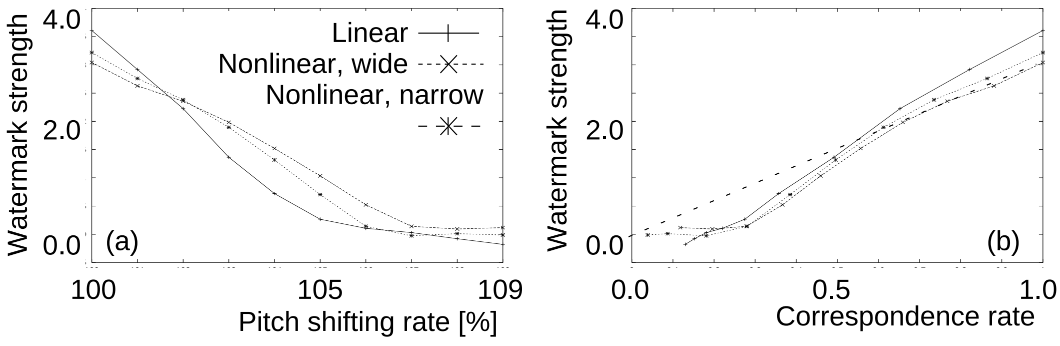

Figure 4.18: The watermark strength detected from pitch-shifted content for subbands with different bandwidths.

A robustness test was conducted using three types of subbands shown in Fig. 4.17. Every subband in the linear subbands shown in Fig. 4.17(b) has the same width of 10 frequency bins. The nonlinear subbands have wider bandwidths in higher frequencies per the following rule: (1) the lowest subband begins at the 10th bin; and (2) the bandwidth is the smallest integer with $r_b[b]$ larger than r_{bmin}, but not less than 3 bins. The value of r_{bmin} is 0.1 for the nonlinear wide subbands (NWS) shown in Fig. 4.17(c), and 0.08 for the nonlinear narrow subbands (NNS) shown in Fig. 4.17(d). While linear and NWS have 30 subbands, NNS is given 34 subbands so that it contains approximately the same number of frequency bins. Figure 4.17(a) shows the relationships of subband numbers versus frequencies for the three designs.

Figure 4.18(a) shows experimental results on the degradation of the average detected bit strengths for these subband designs. Pitch shifting is performed using linear interpolation without anti-alias filtering. Linear subbands mark stronger watermark strengths for the content just after embedding but degrade more rapidly than nonlinear subbands. NWS shows even more robustness than NNS. Figure 4.18(b) shows the same experiment differently. The abscissa is the correspondence rate calculated by Eq. (4.28). It can be seen that the detected watermark strengths are proportional with the correspondence rate as long as the rate is high enough. We believe the influence of the modulus operators, which have opposite values for a frame and for the second frame after the frame, is the reason that the strengths are lower than expected when the pitch-shifting rate is high.

(d) Duration of pattern block

The duration of a pattern block, B_W, is another important parameter. To clarify its influence on the robustness, we implemented four systems using different values of B_W 7, 9, 11, and 13, and examined the degradation of the watermark strength from pitch shifting (Fig. 4.19). The number of tiles for a bit (W_B) is 24, 30, 36 and 42, respectively. Figure 4.19(a) shows that larger B_W results in stronger watermark strengths for the content just after embedding. The strength is proportional to the square root of the number of tiles for a bit (Fig. 4.19(c)), as is theoretically expected. The reason is that the numerator of Eq. (4.22) increases proportionally with the number of tiles, while the denominator increases proportionally with the square root of the number of tiles, and hence the mean of $y[j]$ increases proportionally with the square root.

Furthermore, Figure 4.19(a) shows that larger pattern block duration does not result in better robustness for high pitch-shifting rates. This can be explained as follows: for detecting highly pitch-shifted content using large B_W, mismatches between the original and shifted tiles

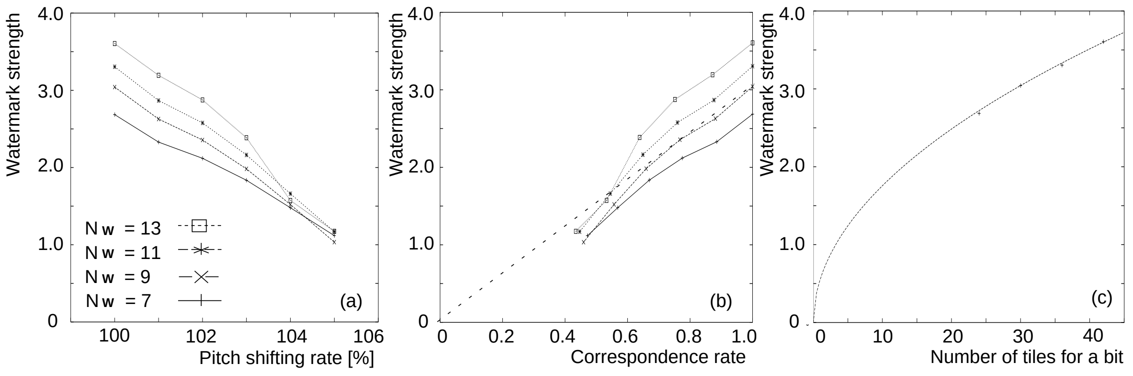

Figure 4.19: The watermark strength detected from pitch-shifted content for different duration of blocks. (c) shows that the original watermark strength increases proportionally with the square root of the number of tiles.

accumulate over the longer duration, so tiles at both ends of the pattern block do not contribute to the watermark detection. Also the effect of the modulus operators occurs earlier. On the other hand, too small a value of B_W leads to a risk of missing a whole pattern block. The duration of pattern blocks should be set considering what sort of degradations the system must be robust against. The relationship of the correspondence rate and the watermark strengths is shown again in Fig. 4.19(b).

(e) Robustness

We tested the robustness of the method against various transformations (Table 4.1). The table shows (1) the means and (2) the variances of the detected watermark strength, (3) the bit error rates (BER), and (4) the rates at which the correct 64-bit message is detected. The watermark strength data for (1), (2) and (3) are taken after the accumulation of twice-repeated codes before the BCH decoding. Hence, the bit errors seen in Table 4.1 can be recovered by the BCH decoding or detected by the CRC parity check. Although a bit error was counted when the accumulated watermark strength is smaller than zero, because the detection algorithm does not output the extracted message if there are too many bits weaker than the threshold (Eq. (4.25)), this incorrect message would normally be eliminated. For the statistical experiment, ten 100-second music samples were used. Since the message is expected to be detected three times in a 100-second music sample, *100%* in Table 4.1 indicates 30 correct detections of the message from the ten samples. In the table, *Original watermark* means no transformation is performed on the content after watermark embedding. *Wow-and-flutter* is a combination of two consecutive transformations: (1) *wow* is a computer simulation of a 0.707% variation of playback speed with a 5 Hz cycle time; and (2) *flutter* is a computer simulation of 0.707% variation of playback speed at a random modulation frequency up to 250 Hz. *Echo 50 msec 0.3* is echoing with maximum delay 50 msec and feedback coefficient 0.5. *Random stretching* is a transformation that modifies the length of the content to the target length by omitting or inserting a random number of samples from 50 up to 500. Random sample cropping can be considered as random stretching with the target length smaller than 100%. *MiniDisc & DAAD* is a combination of recording on a MiniDisc using ATRAC 292 kbps, a digital-to-analog conversion (D/A), and an analog-to-digital conversion (A/D). *DAAD & DAT & DAAD* is a combination of D/A, A/D, recording on a DAT tape, D/A, and A/D. *ATRAC3*

Table 4.1: The means and variances of the detected watermark strength, the bit error rates (BER), and the rates by which the correct 64-bit message is detected. Note that all bit errors were corrected or detected, and hence no wrong message was extracted from the content.

Processing	Mean	Variance	BER	Correct detection
Original watermark	4.23	0.54	0	100%
Wow-and-flutter	3.77	0.66	0.00026	100%
Echo 50m sec 0.3	3.74	0.79	0.00026	100%
MiniDisc & DAAD	3.71	0.67	0.00026	100%
DAAD & DAT & DAAD	3.70	0.77	0.00026	100%
Pitch shifting +2%	3.33	0.78	0.001302	100%
Pitch shifting -2%	3.35	0.80	0.001563	100%
ATRAC3 132kbps & DAAD	2.87	0.78	0.001563	97%
Random stretching +2%	3.39	0.87	0.001823	100%
Noise -40dB	3.49	1.02	0.002083	97%
Random stretching -2%	3.37	0.81	0.002344	100%
Random stretching -4%	2.77	0.92	0.005208	100%
Random stretching +4%	2.88	0.95	0.005729	100%
Echo 100m sec 0.5	2.90	1.02	0.00625	97%
MP3 96kbps	2.48	0.85	0.008333	100%
ATRAC3 105kbps & DAAD	2.54	0.92	0.009896	93%
Pitch shifting +4%	2.25	0.90	0.013021	90%
Pitch shifting -4%	2.32	1.00	0.017448	83%
Noise -30dB	2.51	1.39	0.030208	87%

132 kbps (105 kbps) & DAAD is a combination of encoding by ATRAC3 132 kbps (105 kbps), playback on a SONY MemoryStick Walkman, and recording on a PC.

Correct detection rates over 80% were seen for every one of the tested degradations. The error correction and detection algorithm and the counting of weak bits (Eq. (4.25)) successfully avoided detection of a wrong message.

(f) Discussion

We presented a watermarking method that can embed a 64-bit message onto a 30-second music sample. The method was robust against shift and fluctuation with respect to time and frequency of the audio content. The robustness was achieved mainly through a two-dimensional pseudo-random array, magnitude modification, and nonlinear subbands.

The method modifies the magnitudes of segmented areas in the time-frequency plane of the content according to a two-dimensional pseudo-random array assigned to the areas. Windowing and overlapping were used before and after the DFTs in order to avoid generating clicking sounds at the borders of adjacent DFT frames. It was shown that the effect of the magnitude modification by the embedding algorithm was observable by the detection algorithm with displaced DFT windows. This made the method robust against random cropping without computationally expensive searching for the embedding DFT windows.

It was shown that the correspondence between the embedded watermark and the pseudo-random array used for detection played an important role in determining the detected watermark strength. The watermark strength detected from pitch-shifted content was successfully estimated by a simple calculation of the area maintaining the correspondence. To keep the correspondence rate high, it was better to design subbands having wider bandwidths for higher frequencies.

The duration of the pseudo-random array was also important. The duration can be lengthened so that the loss in some portions of the array can be recovered by the other portions. Actually the watermark strengths increased proportionally with the square root of the length of the array. However it was shown that too long a duration was not effective for robustness against transformations that change the length of the content. We tested the robustness of the method against wow-and-flutter, echo, noise addition, MP3 compression, ATRAC3 compression, MiniDisc recording, random cropping, pitch shifting, digital-to-analog and analog-to-digital conversion.

4.2 Improving Robustness against Geometric Distortion

The robust audio watermarking algorithm introduced in the previous section is robust against pitch shifting and random stretching to some extent. However, it is still difficult for the method to survive excessive geometric distortions. In this section, we explain a modification to the detection algorithm to improve the robustness against excessive distortion. The key features of the modified algorithm are;

- The modified method uses multiple pseudo-random arrays each of which is stretched assuming a certain amount of distortion. When the watermarked audio content is geometrically distorted, watermark can be detected by the pseudo-random array that is stretched in a similar way to the geometric distortion of the audio content.

- A pseudo-random array is chosen from the multiple arrays by the strength of a synchronization signal. Because the strength of the synchronization signal and that of the message signal are independent, the false alarm rate is preserved.

- Because the same synchronization signal, which was necessary for the original detection algorithm to search the head of the message, is used for the scale selection, this method does not decrease the data payload.

- Since most of the detection process for the multiple arrays is shared, the additional computational cost is limited.

4.2.1 Detection using stretched patterns

In the experiment, we used a software system that can embed and detect a 64-bit message in 30-second pieces of music. Its details and parameters that are not explained below are same as explained in Section 4.1. All the following graphs are experimental results using ten 100-second music samples. The watermark strength data plotted in the figures or shown in the table are measured after the accumulation of doubly-encoded watermarks. *Pitch shifting* is performed using linear interpolation without anti-alias filtering. *Random stretching*[2] is a transformation that changes the length of the total content to a different length by omitting or inserting a random number of sample blocks from 50 up to 500 samples per block.

When the content is distorted by pitch shifting or random stretching, the time and frequency location of the embedded tiles are displaced. Accordingly, our idea for improving

[2] *Random sample cropping* can be considered as random stretching with the target length smaller than 100%.

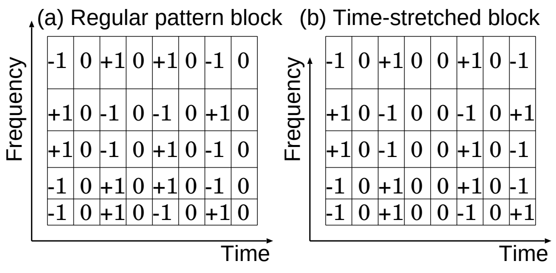

Figure 4.20: Pattern block stretched with respect to time.

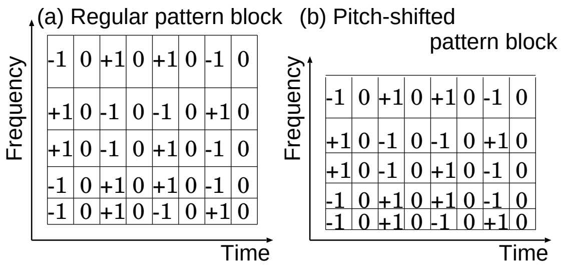

Figure 4.21: Pattern block stretched with respect to frequency.

robustness is to detect watermark using multiple patterns each of which is stretched in advance assuming a certain amount of distortion. As for random stretching, because it changes the length of the content, watermark is expected to be detectable using a pattern that is also stretched with respect to time(Fig. 4.20). The watermark strength using the pattern that is stretched at the rate of r_t is calculated by

$$y[j] = \frac{\sum_{\forall(t,b)\in L} \omega_F[t,b](u_d\left[\lfloor 4r_t t + 0.5\rfloor, b\right] - \overline{u_d})}{\sqrt{\sum_{\forall(t,b)\in L} \{\omega_F[t,b](u_d\left[\lfloor 4r_t t + 0.5\rfloor, b\right] - \overline{u_d})\}^2}}. \tag{4.29}$$

To detect watermark from a sample whose pitch is shifted at the rate of r_f, we correspondingly shift the subbands (Fig. 4.21) as $\text{kmin}[b]' = r_f\text{kmin}[b]$ and $\text{kmax}[b]' = r_f\text{kmax}[b]$. Moreover, since linear pitch shifting changes the duration of the block as well as its frequency, we also stretch the pattern using Eq. (4.29).

In this way, we define a *stretched detector*, $D(r_t, r_f)$, which matches best to a time expansion rate, r_t, and a pitch shifting rate, r_f. Figure 4.22 and Figure 4.23 show mean watermark strengths detected by $D(0.90, 1.00)$, $D(0.935, 1.07)$, $D(1.00, 1.00)$, $D(1.075, 0.93)$, and $D(1.10, 1.00)$ from distorted content. While the strength detected by the regular detector, $D(1.00, 1.00)$, decreases as the content is severely distorted, the stretched detectors have their maximum strengths approximately at their assumed distortion rates.

4.2.2 Experimental results

These experiments indicate that if several stretched detectors detect watermark in the music sample using differently stretched patterns in parallel, and an appropriate stretched detector is selected, we can detect watermark even from an excessively distorted music sample by some of the stretched detectors. In this way, the detection flow becomes as shown in Fig. 4.24. The selection of a stretched detector is done approximately every 30 seconds based on the

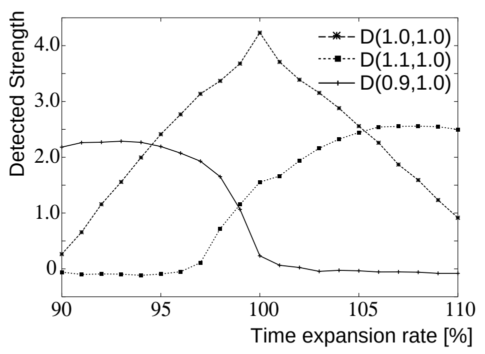

Figure 4.22: Mean strengths detected from randomly stretched samples.

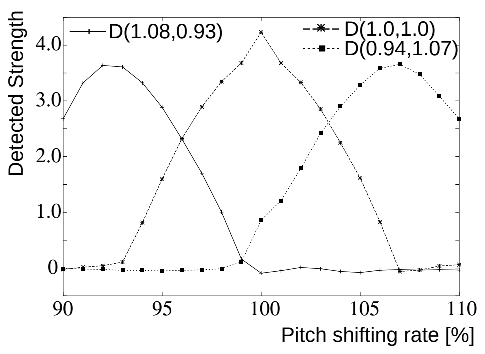

Figure 4.23: Mean strengths detected from pitch-shifted samples.

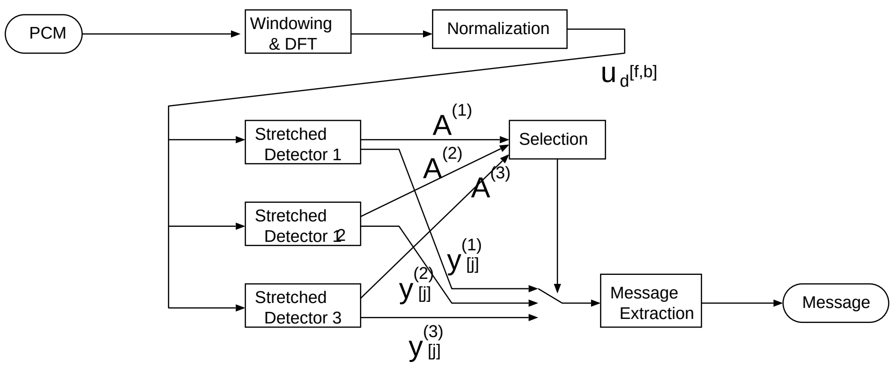

Figure 4.24: Detection flow with three stretched detectors.

accumulated synchronization strength,

$$A^{(i)} = \frac{1}{\sqrt{N_{b30}}} \sum_{q=1}^{N_{b30}} S^{(i)}[q], \tag{4.30}$$

where i is the index of stretched detectors, n is the index of synchronization signals detected in the 30-second period, and N_{b30} is the number of synchronization signals detected in the period. After the stretched detector that gives the maximum $A^{(i)}$ is selected, the watermark strengths detected from the stretched detector are used for the message reconstruction.

The mean of $A^{(i)}$ detected from five stretched detectors are shown in Fig. 4.25 and Fig. 4.26. It can be seen in Fig. 4.26 that, for example, $D(1.08, 0.93)$ is selected for the pitch-shifting rate ranging from 90% up to 96%.

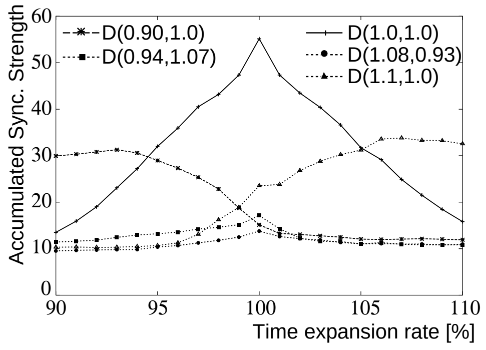

Figure 4.25: Mean accumulated sync. strength detected from randomly stretched music samples.

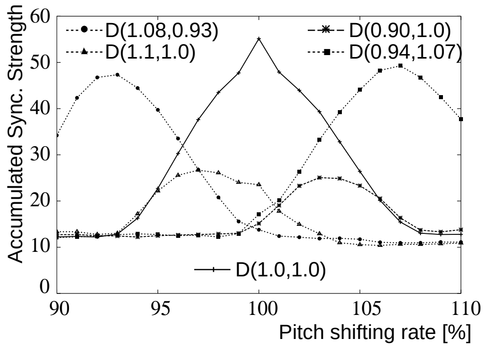

Figure 4.26: Mean accumulated sync. strength detected from pitch-shifted music samples.

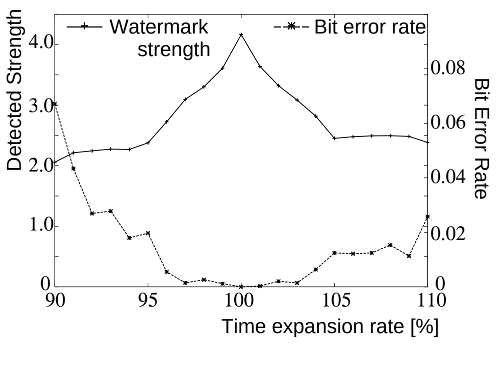

Figure 4.27: Mean strength and bit error rate (BER) for randomly stretched samples.

Consequently, the mean strengths by the selected stretched detectors become as shown in Fig. 4.27 and Fig. 4.28, and are enough high for every degree of tested distortion. Corresponding bit error rates are also shown in the figures.

Table 4.2 shows (1) the means of the detected strengths, (2) the bit error rates (BER) which are plotted in Fig. 4.27 and Fig. 4.28, and (3) the correct detection rates (CDR) at which correct 64-bit message was detected. CDR over 80% was seen for every one of the tested degradation. The error correction and detection algorithm and the counting of weak bits successfully avoided detection of an incorrect message.

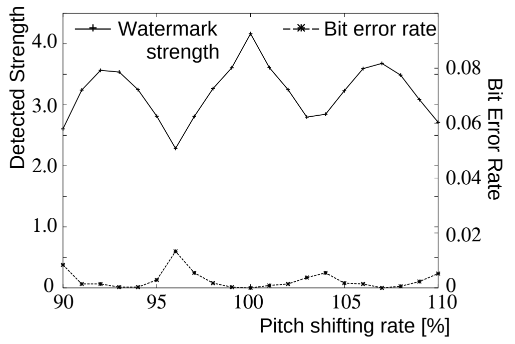

Figure 4.28: Mean strength and BER for pitch-shifted samples.

Table 4.2: The means of the detected strengths (μ), the bit error rates (BER), and the correct detection rates (CDR).

Processing	μ	BER	CDR
Original watermark	4.17	0.000	100%
Pitch shifting -10%	2.61	0.008	96%
Pitch shifting -8%	3.56	0.001	100%
Pitch shifting -6%	3.25	0.000	100%
Pitch shifting -4%	2.29	0.013	90%
Pitch shifting -2%	3.27	0.002	100%
Pitch shifting +2%	3.25	0.001	100%
Pitch shifting +4%	2.85	0.005	100%
Pitch shifting +6%	3.59	0.001	100%
Pitch shifting +8%	3.49	0.001	100%
Pitch shifting +10%	2.71	0.005	100%
Random stretching -10%	2.06	0.067	83%
Random stretching -8%	2.25	0.027	87%
Random stretching -6%	2.27	0.018	87%
Random stretching -4%	2.72	0.005	100%
Random stretching -2%	3.30	0.003	100%
Random stretching +2%	3.32	0.002	100%
Random stretching +4%	2.82	0.006	100%
Random stretching +6%	2.48	0.012	93%
Random stretching +8%	2.49	0.015	93%
Random stretching +10%	2.38	0.026	87%

4.2.3 Performance

We also measured the detection speed using a personal computer (PC) with 600 MHz Pentium III running Windows NT. While, when the detector uses only a regular detector, detection takes 7.72% of the length of the content, when the detector uses five stretched detectors, it is performed within 8.96% of the length of the content. That is only 16% increase for four more stretched detectors. This is because the Fourier transform and calculation of the normalized magnitudes take most of the processing time, and the stretched detectors can share this part of calculation. Therefore, using stretched detectors in a detector is much faster than simply using multiple detectors.

We improved robustness of the basic watermarking algorithm by using multiple stretched pattern blocks. With the improvement, robustness against pitch shifting and random stretching up to ±10% was achieved with only 16% additional computational time.

4.3 Capacity Analysis

In this section, we analyze the communication capacity of the algorithm described in Section 4.1 under the condition where additive noise is overlapped. The communication capacity of audio watermarking is the amount of data payload that can be embedded into audio content while preserving acoustic naturalness and keeping robustness against distortions. Robustness against additive noise is a simplest but practical requirement necessary for automatic monitoring of broadcast music. The capacity analysis is done by taking into consideration the characteristic of natural audio signal having more energy in lower frequencies.

In the following sections, after simplified formulation of watermark detection is introduced, a watermark-signal-to-noise ratio (WMSNR) is defined in Section 4.3.1. Then the relationship between WMSNR and the communication capacity is derived. When additive noise is overlapped, WMSNR for the content degrades. The degradation of WMSNR against additive noise is theoretically estimated in Section 4.3.2. The validity of the estimation is verified by experimental results in Section 4.3.3. By examining the experimental results, the communication capacity of the algorithm is analyzed. Section 4.3.4 is an appendix section for detailed equations for the derivation of WMSNR.

4.3.1 Relationship between watermark SNR and capacity

Preparation In preparation for deriving the relationship, we simplify watermark detection equations. First, we approximate Eq. (4.14) by $\widetilde{c_a}[f,k] = \log c_a[f,k]$. This is because normalization by the average magnitude of the frame does not have much practical effects as long as logarithmic magnitudes are used. Then, we rewrite Eq. (4.15) by the following equation assuming that every subband has the same number F_H of coefficients.

$$u[f,b] = \sum_{k=\mathrm{kmin}[b]}^{\mathrm{kmax}[b]} \widetilde{c_a}[f,k]. \tag{4.31}$$

The tile value $u_d[f,b]$ is calculated by Eq. (4.16). The detected bit strength is calculated as the simply correlation between the pseudo-random sequence and the tile values.

$$y[j] = \sum_{\forall(t,b)\in L[j]} \omega_F[t,b]u_d[f,b], \tag{4.32}$$

in which we simplified Eq.(4.22) by omitting the denominator. This does not affect the estimated communication capacity because the denominator was originally used to control the variance of the bit strength and because we instead factor the variance into the calculation by considering the variance of the bit strength. In addition, $\overline{u_d}$ in the original equation is approximated by 0. f is an index of a frame included in the t-th column of the block. For example, $4t$ can be used as f if we assume the synchronization position is at the 0-th frame.

Although the following calculations are not necessary for the detection steps, we define some additional variables for later use. $r[f,k]$ is multiplication of a pseudo-random number and a normalized magnitude coefficient. $h[t,k]$ is the difference between the multiplication values for adjacent frames.

$$\begin{aligned} r[f,k] &= \omega_F[t,b]\widetilde{c_a}[f,k], & (4.33)\\ h[t,k] &= r[f,k] - r[f+2,k], & (4.34)\end{aligned}$$

where k is in the b-th subband ($\mathrm{kmin}[b] \le k \le \mathrm{kmax}[b]$). By applying Eq.(4.31) and Eq.(4.16) to Eq. (4.32), we can derive the following relationship.

$$\begin{aligned} y[j] &= \sum_{\forall(t,b)\in L[j]} \omega_F[t,b]u_d[f,b] & (4.35)\\ &= \sum_{\forall(t,b)\in L[j]} \omega_F[t,b](u[f,b]-u[f+2,b]) & (4.36)\\ &= \sum_{\forall(t,b)\in L[j]} \omega_F[t,b]\left(\sum_{k=\mathrm{kmin}[b]}^{\mathrm{kmax}[b]} \widetilde{c_a}[f,k] - \sum_{k=\mathrm{kmin}[b]}^{\mathrm{kmax}[b]} \widetilde{c_a}[f+2,k]\right) & (4.37)\\ &= \sum_{\forall(t,b)\in L[j]} \sum_{k=\mathrm{kmin}[b]}^{\mathrm{kmax}[b]} (r[f,k]-r[f+2,k]) & (4.38)\\ &= \sum_{\forall(t,b)\in L[j]} \sum_{k=\mathrm{kmin}[b]}^{\mathrm{kmax}[b]} h[t,k], & (4.39)\end{aligned}$$

where the number of accumulated $h[t,k]$s is $W_B F_H$.

Derivation of the relationship We assume the bit strength y follows a Gaussian distribution with the mean $\mathbf{E}[Y]$ and the variance $\mathrm{Var}[Y]$. Then, the probability of a bit error can be restated as the probability of the tail of the distribution cross the origin, $Q(\sqrt{\mathbf{E}[Y]^2/\mathrm{Var}[Y]})$, where $Q(x)$ is the cumulative distribution function defined by the following equation,

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-t^2/2} dt. \tag{4.40}$$

$\mathbf{E}[Y]^2/\mathrm{Var}[Y]$ is a important parameter that determines the capacity of communication through watermark. We name the parameter a watermark signal-to-noise ratio (WMSNR),

$$\mathrm{WMSNR}_B = \frac{\mathbf{E}[Y]^2}{\mathrm{Var}[Y]}. \tag{4.41}$$

By Eq. (4.39), we can derive $\mathbf{E}[Y] = W_B F_H \mathbf{E}[H]$ and $\mathrm{Var}[Y] = W_B F_H \mathrm{Var}[H]$. Hence, Eq. (4.41) can be rewritten as

$$\mathrm{WMSNR}_B = W_B F_H \frac{\mathbf{E}[H]^2}{\mathrm{Var}[H]}. \tag{4.42}$$

We define WMSNR per frequency component by

$$\mathrm{WMSNR}_C = \frac{\mathbf{E}[H]^2}{\mathrm{Var}[H]}. \tag{4.43}$$

Shimizu compared some watermark embedding methods and communication methods in [61] to estimate a communication capacity that can be achieved under a defined level of degradation with an error rate below a predefined threshold. For example, the probability P_M of error that a wrong codeword is decoded for Direct Sequence Spread Spectrum (DSSS) without coding can be estimated with the following equation.

$$\begin{aligned} P_M &= 1-(1-P_b)^R \approx RP_b \quad (4.44) \\ &= RQ\left(\sqrt{2\frac{S}{N}\frac{W}{R}}\right), \quad (4.45) \end{aligned}$$

where R, P_b, and W are the communication capacity, the bit error rate, and the bandwidth, respectively. We can obtain the communication capacity R with a predefined error rate P_M for a certain S/N by solving the equation adversely. Note that the S/N in the Shimizu's equation corresponds to our WMSNR$_C$ defined in Eq. (4.43) [3]. In the following section, by calculating the WMSNR for each level of degradation, we estimate the communication capacity by using the Shimizu's theory.

4.3.2 Estimation of watermark SNR

In this section, we derive an equation that estimates the WMSNR for a given level of degradation of the content. With the estimated WMSNR, we can estimate the communication capacity as well by using Eq. (4.45) or similar equations for other communication methods. Before deriving the WMSNR estimation equation, we describe our assumptions and some losses we particularly take into consideration.

(a) Assumptions

Assumption on the host signal We assume the host signal locally follows a Gaussian distribution while the variance of the distribution is different depending on the portions of the host signal. In other words, from a local point of view, the real components $c_r[k]$ and the imaginary components $c_i[k]$ of the host signal mutually independently follows a Gaussian distribution with the mean 0 and the variance σ_r^2. In addition, we assume the amplitude frequency components of a frame (f) and a next non-overlapping frame ($f+2$) have correlation. These assumptions are based on the fact that musical signals sustain a similar stochastic characteristic for a certain length of time. However, from a global point of view, different portions of the content have different values of σ_r^2 and the distribution of σ_r^2 is dependent on the genre of the content.

Assumption of simple psychoacoustic model We assume the use of a simplest psychoacoustic model that determines the magnitude of the watermark signal in proportion to the magnitude of the host signal as $a_p[f,k] = \beta c_a[f,k]$, where β is a constant value. Assuming this model, the purpose of the embedding algorithm is restated as modifying the magnitudes as determined by the following equation:

$$\begin{aligned} {c_a}'[f,k] &= c_a[f,k] + s[f,b]a_p[f,k] \quad (4.46) \\ &= (1+s[f,b]\beta)\, c_a[f,k], \quad (4.47) \end{aligned}$$

where k is include in the b-th subband ($\mathrm{kmin}[b] \le k \le \mathrm{kmax}[b]$). This operation also multiplies both the real component $c_r[f,k]$ and the imaginary component $c_i[f,k]$ by $(1+s[f,b]\beta)$ and makes ${c_r}'[f,k]$ and ${c_i}'[f,k]$.

[3]In the Shimizu's equation, S/N is defined as $\frac{S}{N} = \frac{\mu_s^2}{\sigma_s^2+\sigma_N^2}$, where μ_s and $\sigma_s^2+\sigma_N^2$ in the equation correspond to $\mathbf{E}[H]$ and $\mathrm{Var}[H]$ of Eq. (4.43), respectively.

Of course, more complex psychoacoustic model taking masking effects into consideration should be used in practice. However, it is very difficult to analyze the communication capacity considering the masking effects since the magnitudes of the frequency components mutually affect the magnitudes of the watermark signals at different frequencies. In addition, there is no commonly used standard psychoacoustic model. These are the reason we use the simplest psychoacoustic model for analysis.

Assumption on degradation We assume as a model of degradation that a Gaussian noise signal is added to each of the real components and the imaginary components of the watermarked signal by

$$c_r''[f,k] = c_r'[f,k] + \nu_{F,r}[f,k], \quad (4.48)$$
$$c_i''[f,k] = c_i'[f,k] + \nu_{F,i}[f,k], \quad (4.49)$$

where $c_r''[f,k]$ is the k-th real component of the watermarked and degraded signal for f-th frame. We assume the noise signal $\nu_{F,r}[k]$ follows a Gaussian distribution, $N(0, \sigma_\nu^2)$. In addition, we assume that there is no correlation between the noise signals of a frame and the noise signal of the neighboring frames and that there is no correlation between the magnitudes of the host signal and the magnitudes of the noise signal. We assume the same thing for the imaginary component c_i''.

(b) Loss by the windowing function

Although the embedding algorithm is intended to increase or decrease the host signal by the watermark signal whose magnitude is β times of that of the host signal, the actual change of the magnitude is diminished by the use of the windowing function. However, this loss is unavoidable because the windowing function is necessity to make detection possible even when the detection DFT frames are displaced from the original embedding DFT frames.

Observable change of magnitudes The watermark signal is multiplied by the windowing function when the step 6 of embedding performs inverse DFTs (IDFTs) and when the step 1 of detection performs DFTs. By these processes of mulutiplication of the windowing function, the change of magnitudes observed by the detection algorithm is approximately only $\frac{3}{4}\beta$ even though the magnitudes are changed by the rate of β. We denote the coefficient $\frac{3}{4}$ as γ.

Increase of variances by the windowing function If $\widetilde{c_a}[f,k]$s in a same subband in Eq. (4.15) are mutually independent, the mean and the variance of $u[f,b]$ will respectively become F_H-times of those of $\widetilde{c_a}[f,k]$. However, the use of a windowing function for observation of the magnitudes causes correlation between the neighboring magnitudes and results in larger values for the variance of u. We found by simulation using MATLAB[4] that the correlation coefficient of the adjacent logarithmic magnitude components has a value of $\rho_B \approx 0.159$ when a sine window is used for the windowing function, even though the host signal is a Gaussian signal. This effect makes the variance of u, $\left(1 + 2\frac{F_H-1}{F_H}\rho_B\right) = 1.28$ times. We denote the value 1.28 as ξ.

(c) Estimation of WMSNR

In the following section, we first make an equation to estimate WMSNR in a local region where we can consider the standard deviation σ_r of the host signal constant. Then, based on the

[4]A numerical computing environment developed by MathWorks.

equation, we make a global estimation equation of WMSNR that takes the distribution of σ_r into consideration.

Estimation of local WMSNR Without limiting the generality of the discussion, we can confine the discussion to the j-th bit whose value m_j is 1. According to Eq. (4.7), the sign $s[f, b]$ for the first two frames of each tile equals to the pseudo-random number $\omega_F[t, b]$. We hereafter abbreviate the suffix $[t, k]$ in Eq. (4.39) by i. Instead, we distinguish the f-th frame and the $(f + 2)$-th frame by using the suffices 1 and 2, respectively. The number of frequency components to be summed is $W_B F_H$. Based on these definitions, Eq. (4.39) can be expressed by

$$\begin{aligned} y[j] &= \sum_{i=0}^{W_B F_H - 1} h[i] = \sum_{i=0}^{W_B F_H - 1} (r_1[i] - r_2[i]) && (4.50) \\ &= \sum_{i=0}^{W_B F_H - 1} \omega_F[i](\log c_{a1}[i] - \log c_{a2}[i]). && (4.51) \end{aligned}$$

Furthermore, we can derive the following equation as shown in Section 4.3.4.

$$\begin{aligned} \mathbf{E}[H] &= \frac{1}{2} \log \left\{ \frac{(1 + \gamma\beta)^2 + d^2}{(1 - \gamma\beta)^2 + d^2} \right\}, && (4.52) \\ \mathrm{Var}[H] &\approx 2 \times 0.411(1 - \rho_L). && (4.53) \end{aligned}$$

The loss γ by the windowing function is taken into consideration. Note that $d = \sigma_\nu / \sigma_r$. ρ_L is the correlation coefficient of the logarithmic magnitudes, $\log c_{a1}$ and $\log c_{a2}$, of adjoining frames. If the host signal is music, ρ_L has a non-zero value. We can estimate WMSNR$_C$ by using this to Eq. (4.43) and factoring the increase of the variance by the windowing function into the calculation.

$$\begin{aligned} \mathrm{WMSNR}_C &= \frac{\mathbf{E}[H]^2}{\xi \mathrm{Var}[H]} && (4.54) \\ &\approx \frac{\log \left\{ \frac{(1+\gamma\beta)^2 + d^2}{(1-\gamma\beta)^2 + d^2} \right\}}{4 \times 0.411 \xi (1 - \rho_L)}. && (4.55) \end{aligned}$$

We can see the use of the correlation coefficient ρ_L, by taking differences of adjoining frames, increases WMSNR$_C$ by a factor $\frac{1}{1-\rho_L}$.

Estimation of global WMSNR The local estimation of WMSNR derived in the previous section is not practically useful since it assumes the host signal follows a unique distribution. Contrarily, the distribution of the host signal differs for each audio content. Since degradation of watermark begins at portions of the audio content whose magnitudes are small, the difference of the distribution has a substantial impact on the watermark robustness. To take the distribution difference into consideration, we investigate the probability density function, $P_v(\sigma_r)$, of the standard deviation, σ_r, of the magnitude coefficients of each audio content. Using $P_v(\sigma_r)$, we estimate the global WMSNR$_C$ for the whole audio content by

$$\begin{aligned} &\mathrm{WMSNR}_C \\ &= \left\{ \int_0^\infty P_v(\sigma_r) \sqrt{\frac{E[H]^2}{Var[H]}} d\sigma_r \right\}^2. && (4.56) \end{aligned}$$

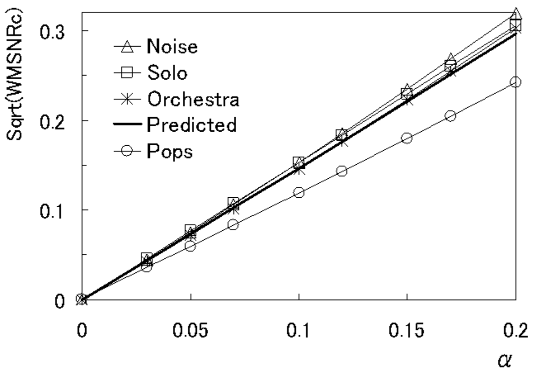

Figure 4.29: Experimental results of $\sqrt{\text{WMSNR}_C}$ estimation. The lines except the line for *Estimated* are experimentally observed results.

4.3.3 Experimental results

In this section, we first compare the estimated WMSNR and the experimentally observed WMSNR for the cases without additive noise to verify the estimation equation. Then we conduct a similar comparison for WMSNR after noise addition. We draw a conclusion in the last part of the section.

(a) Parameter values

We used ten pieces of monaural host signals for the experiments; three pieces from each of orchestral music, pop music, and solo instruments (piano, violin, and harpsichord) and one piece of white noise. All of the signals were sampled at a frequency of 44.1 kHz and with a bit resolution of 16 bits, and each piece is 100 seconds long. The distribution of the standard deviations of magnitudes was investigated for each of the four genres. The lowest frequency kmin[1] for the lowest subband was set to the 10-th frequency component. The number of subbands B_H was 24, which corresponds to the range from 400 Hz to 11 kHz. Each of the 24 subbands was given $F_H = 10$ frequency components. The length B_W of a pattern block was set to 10. Since we used the whole block for a bit $N_b = 1$, the number of tiles for a bit was $W_B = B_W B_H = 240$, $W_B F_H = 2,400$. With the length of a DFT frame $N = 1,024$, the length of a block corresponds also to $N/2 \times 4 \times B_W = 20,480$ samples ≈ 0.464 seconds.

(b) WMSNR without noise addition

We compared the estimated values (Eq.(4.54)) and the experimentally observed values of WMSNR$_C$ just after embedding without noise addition in Fig. 4.29, whose horizontal axis and vertical axis are β and square roots of WMSNR$_C$, respectively. The graph legends are sorted in descending order. The estimated values are calculated assuming that the host signal is a Gaussian noise and that the magnitude components are distributed in a Rayleigh distribution with the correlation coefficient ρ_L 0. No additive noise is added ($\sigma_\nu = 0$).

(c) WMSNR after noise addition

We also compared the estimated values (Eq.(4.56)) and the experimentally observed values of WMSNR$_C$ after noise addition. We used the probability density functions (Fig. 4.30) that we obtained by actually investigating the test audio content for $P_v(\sigma_r)$ of Eq. (4.56). The

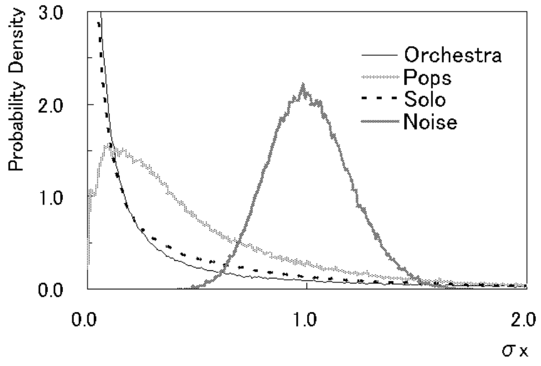

Figure 4.30: Local distributions of magnitudes for each genre.

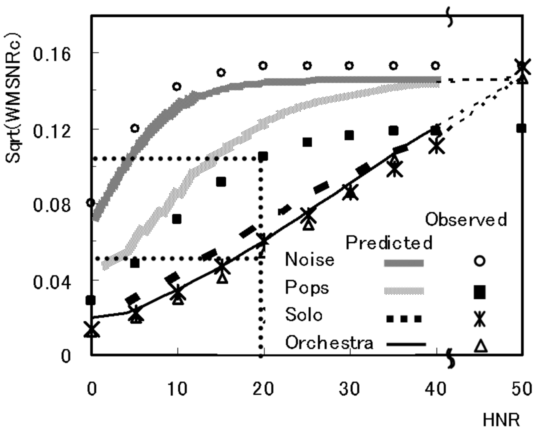

Figure 4.31: Estimated and observed $\sqrt{\text{WMSNR}_C}$ against changing global host-signal-to-additive-noise ratio(HNR).

horizontal axis of the graph is normalized by the average energy calculated for entire samples in each genre.

The energy of additive noise was controlled to be constant for each of the test audio files and to be in a fixed ratio compared to the average energy of the audio file. In other words, additive noise that results in a certain amount of signal to noise ratio (host signal to additive noise ratio) was added. This is done simulating analogue transmission such as television transmission and radio transmission. The host signal to additive noise ratio (HNR) when σ_ν is added can be expressed in the following equation:

$$\text{HNR} = 10\log_{10}\int_0^\infty P_v(\sigma_r)\left(\frac{\sigma_r}{\sigma_\nu}\right)^2 d\sigma_r. \tag{4.57}$$

Figure 4.31 shows the estimated values and the experimentally observed values of $\sqrt{\text{WMSNR}_C}$ against various values of HNR$[dB]$. 30 of the horizontal axis means that the energy of the additive noise signal is -30 dB compared to the energy of the host signal. The data plotted at the right edge of the figure are values before noise addition. $\beta = 0.1$ was used for watermark embedding.

(d) Discussion

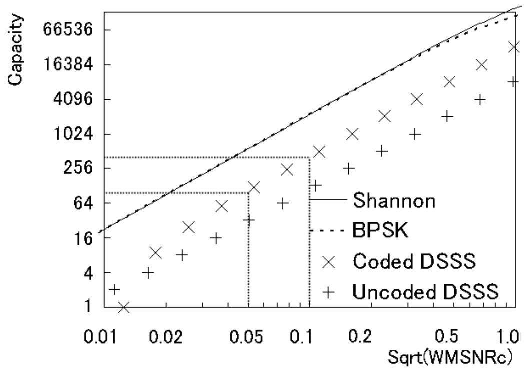

Figure 4.32: Watermark capacity with the bandwidth 155,039 and the error rate 1e-5.

Validity of estimation We can see the validity of the estimated WMSNR just after watermark embedding in Fig. 4.29. Change of WMSNR by additive noise is also successfully estimated (Fig. 4.31). Although the estimated values for pop music was somewhat different from the observed values, this is caused by the error of the estimated WMSNR_C just after watermark embedding. We can consider the degradation process is estimated well.

Watermark capacity We can calculate the capacity of audio watermark based on the experimental results as follows; Since there were $W = W_B F_H = 2,400$ frequency coefficients for each pattern block whose length is $2NB_W$ samples ($\approx$ 0.464 seconds), the number of frequency components that are available for detecting watermark from 30-second monaural music is 155,039 [5]. We can obtain the relationship (Fig. 4.32) between signal-to-noise ratio (SNR) and capacity by redrawing the Shimizu's graph with the bandwidth 155,039. While the vertical axis of the figure is the communication capacity with the error rate 10^{-5}, the horizontal axis is $\sqrt{\mathrm{WMSNR}_C}$. According to Fig. 4.31, when additive noise with the HNR 20dB is added, $\sqrt{\mathrm{WMSNR}_C}$ for pop music and orchestral music was approximately 0.1 and 0.05, respectively. By reading these values in Fig. 4.32, we can see that the maximum communication capacity is approximately 400 and 100, respectively. These are the numbers for watermark embedded with $\beta = 0.1$ into 30-second audio pieces.

Dependency on music genres The fact that the robustness and the communication capacity are heavily dependent on the genre of the host signal is attributable to the different distributions of frequency components. Orchestral music and solo instruments have a wide dynamic range of magnitudes. Audio watermark embedded in low-energy portions degrades easily with small quantity of additive noise, while embedding a strong watermark in these portions should be avoided because the low-energy host signal in these portions cannot mask the strong watermark. For these reasons, watermarking orchestral music or solo instruments is more difficult than watermarking pop music.

4.3.4 Derivation of watermark SNR

In this section, Eq. (4.52) is derived. Based on Eq. (4.47) and Eq. (4.48), the real components after watermark embedding and noise addition can be expressed as $c_r''[i] = (1+\omega_F[i]\beta)c_r[i] + \nu_{F,r}[i]$. Since the distributions of $c_r[i]$ and $\nu_{F,r}[i]$ are $N(0,\sigma_r)$ and $N(0,\sigma_\nu)$, respectively, the distribution of $c_r''[i]$ becomes a Gaussian distribution $N(0,(1+\omega_F[i]\beta)^2\sigma_r^2+\sigma_\nu^2)$. The same

[5] $(30 \times 44100)/(N/2 \times 4 \times Bw) \times W = 155,039.0625$.

thing can be said to the imaginary components. In general we can say that, when stochastic variables x and y independently follow a Gaussian distribution $N(0, \sigma^2)$, the mean and the variance of $z = \log \sqrt{x^2 + y^2}$ is $0.0580 + \log \sigma$ and 0.411, respectively. This is calculated by numerical integral with Mathematica, which is a mathematical software system developed by Wolfram Research. Because of the results of the above-mentioned numerical integral, we can know the mean and the variance of $\log c_a''[i]$ are $0.0580 + \log \sqrt{(1 + \omega_F[i]\beta)^2\sigma_r^2 + \sigma_\nu^2}$ and 0.411, respectively. Based on these values, $\mathbf{E}[H]$ and $\mathrm{Var}[H]$ can be obtained by

$$\begin{aligned}
\mathbf{E}[H] &= \mathbf{E}[\omega_F[i](\log c_{a1}''[i] - \log c_{a2}''[i])] &(4.58)\\
&= \frac{1}{2}\left\{(+1 \times \mathbf{E}[\log c_{a+}'' - \log c_{a-}'']) + (-1 \times \mathbf{E}[\log c_{a-}'' - \log c_{a+}''])\right\} &(4.59)\\
&= \mathbf{E}[\log c_{a+}'' - \log c_{a-}''] &(4.60)\\
&= \log \sqrt{(1+\beta)^2\sigma_r^2 + \sigma_\nu^2} - \log \sqrt{(1-\beta)^2\sigma_r^2 + \sigma_\nu^2} &(4.61)\\
&= \log \frac{\sqrt{(1+\beta)^2\sigma_r^2 + \sigma_\nu^2}}{\sqrt{(1-\beta)^2\sigma_r^2 + \sigma_\nu^2}} &(4.62)\\
&= \frac{1}{2}\log \frac{(1+\beta)^2 + \left(\frac{\sigma_\nu}{\sigma_r}\right)^2}{(1-\beta)^2 + \left(\frac{\sigma_\nu}{\sigma_r}\right)^2}. &(4.63)
\end{aligned}$$

The transformation from Eq. (4.58) to Eq. (4.59) separated terms of $+1$ and terms of -1 by considering the fact that ω_F consists of the same number of $+1$s and -1s. c_{a+} and c_{a-} are magnitudes the s for which is positive and negative, respectively.

$$\begin{aligned}
\mathbf{E}[H^2] &= \mathbf{E}[\omega_F[i]^2(\log c_{a1}''[i] - \log c_{a2}''[i])^2] &(4.64)\\
&= \mathbf{E}[(\log c_{a+}'' - \log c_{a-}'')^2] &(4.65)\\
&= \mathbf{E}[(\log c_{a+}'')^2 + (\log c_{a-}'')^2 - 2\log c_{a+}'' \log c_{a-}''] &\\
&\approx (0.411 + \mathbf{E}[\log c_{a+}'']^2) + (0.411 + \mathbf{E}[\log c_{a-}'']^2) &\\
&\quad -2(\mathbf{E}[\log c_{a+}'']\mathbf{E}[\log c_{a-}''] + \rho_L\sqrt{\mathrm{Var}[\log c_{a+}'']\mathrm{Var}[\log c_{a-}'']}) &(4.66)\\
&= 2 \times 0.411(1-\rho_L) + (\mathbf{E}[\log c_{a+}''] - \mathbf{E}[\log c_{a-}''])^2, &(4.67)\\
\mathrm{Var}[H] &= \mathbf{E}[H^2] - \mathbf{E}[H]^2 &(4.68)\\
&\approx 2 \times 0.411(1-\rho_L). &(4.69)
\end{aligned}$$

The transformation from Eq. (4.64) to Eq. (4.65) is not dependent on the value of the pseudo-random number. This is because either $c_{a1}''[i]$ or $c_{a2}''[i]$ must be c_{a+}'' and the other one must be c_{a-}''.

4.4 Concluding Remarks

We presented a watermarking method that can embed a 64-bit message onto a 30-second music sample. The method was robust against shift and fluctuation with respect to time and frequency of the audio content. The robustness was achieved mainly through a two-dimensional pseudo-random array, magnitude modification, and nonlinear subbands.

The method modifies the magnitudes of segmented areas in the time-frequency plane of the content according to a two-dimensional pseudo-random array assigned to the areas. Windowing and overlapping were used before and after the Discrete Fourier Transforms (DFTs) in order to avoid generating clicking sounds at the borders of adjacent DFT frames. It was shown that the effect of the magnitude modification by the embedding algorithm was observable by the detection algorithm with displaced DFT windows. This makes the method robust against random cropping without computationally expensive searching for the embedding DFT windows. Furthermore, we improved robustness of the algorithm by using multiple stretched pattern blocks. With the improvement, robustness against pitch shifting and random stretching up to ±10% was achieved with only 16% additional computational time.

We tested the robustness of the method against wow-and-flutter, echo, noise addition, MPEG1 Audio Layer 3 (MP3) compression, ATRAC3 compression, MiniDisc recording, random cropping, pitch shifting, digital-to-analog and analog-to-digital conversion. Although we do not have results of formal subjective listening tests to present in this chapter, the robustness experiments were performed with the watermark signal level 35dB lower than the level of the host signal, which was the level virtually indistinguishable to human ears. The acoustic quality of the method was verified also through a number of subjective listening tests. In the tests, professional sound engineers and audio equipment critics carefully listened to watermarked audio samples.

The last part of this chapter theoretically formulated the communication capacity of the algorithm under the condition where additive noise is overlapped. It was revealed that the different distributions of magnitudes depending on the genre of audio content caused different robustness against the additive noise.

Further improvement of the algorithm is required to achieve better robustness against excessive distortions and to shorten the duration of content required to carry a message.

Chapter 5

Applications of Robust Audio Watermarking

We extend the applications of the audio watermarking algorithm to a broader range of situations in this chapter, while conventional audio watermarking research assumes limited uses of digital audio data. First, we show that the algorithm described in Chapter 4 can be applied to compressed audio and that the embedded information can be detected whether or not the watermarked audio data is compressed. Second, we describe multiple composition methods for real-time watermark embedding for analogue audio and live performances played in auditoriums, and we point out their merits and flaws. Sonic watermarking, which is one of the composition methods, is a method that can embed watermarks into the sound in the air by making the watermark sound enter the air from the speaker and mixing it with the host sound in the air. In the last part of the chapter, we carefully consider the application model and the possible problems of sonic watermarking and report the experimental results. This chapter is related to the work published in [65, 47, 66, 69, 68, 49, 48, 50].

5.1 Audio Watermarking for MPEG AAC Audio

The main purpose of this section is to show the basic algorithm described in Section 4 can be applied to compressed audio to enable blind watermark detection in AAC[1]-compressed audio content even when the content has been compressed *after* the watermark was embedded in the uncompressed domain. The whole target of this section is illustrated in Fig. 5.1. That will allow watermark detection in both the compressed domain and the uncompressed domain regardless of the original domain where the watermark embedding was done. The most difficult challenge among the paths considered in the figure is watermark detection in the compressed domain when the content is first watermarked in the compressed domain, then decompressed to the uncompressed domain, and finally compressed again. This is because the Modified Discrete Cosine Transform (MDCT) frames used for the second compression are not necessarily identical to the original MDCT frames of the first compressed audio. They may be displaced because of editing such as trimming that was performed after the embedding and the decompression before the compression and the detection (Fig. 5.2). A two-dimensional pseudo-random pattern of the basic algorithm is also effective for solving the problem. Detection can be performed with neither reference to the host signal nor any side information other than the MDCT coefficients. However, to enable detection in the compressed domain, the data payload and the acoustic quality are sacrificed to some extent. The data payload of the method is much less than the

[1]MPEG2 Advanced Audio Coding.

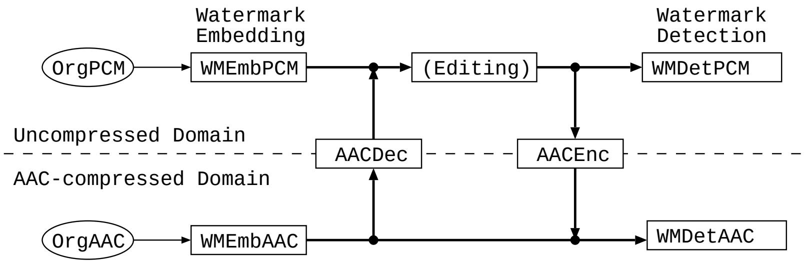

Figure 5.1: The goal of this section is to enable watermark detection in both the compressed domain and the uncompressed domain regardless of the original domain where the watermark embedding was done. The abbreviation are described in Table 5.1.

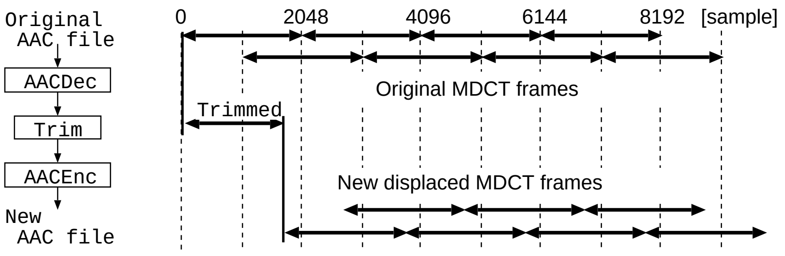

Figure 5.2: When an AAC file is decompressed and compressed again, the MDCT frames may be displaced because of the editing such as trimming.

methods of Neubauer et al. [51, 52, 53, 54]. The acoustic quality of the proposed method has not been tested.

After the key ideas for the method are described in Section 5.1.1, the rest of the section covers the changes necessary for the basic algorithm. The experimental results of magnitude modification and observation demonstrate the effect of the regularity in the pseudo-random pattern in Section 5.1.2. The robustness of the audio watermark embedded in the uncompressed domain or in the AAC-compressed domain is tested in Section 5.1.3.

5.1.1 Algorithms for AAC-compressed content

In this section, after the basic ideas of the method are described, we present the changes necessary for the basic algorithm. The main ideas of the method can be briefly summarized as follows:

- The uncompressed domain watermarking algorithms overlap the Discrete Fourier Transform (DFT) frames with the adjacent frames by a half window just as the MDCT frames of AAC do.

- The statistical non-uniformity of the magnitudes of the content, which has been introduced by the watermark embedding, can be observed from either the DFT coefficients calculated from an uncompressed version of the content or from the MDCT coefficients recovered from an AAC-compressed version of the content.

- The statistical non-uniformity can be introduced into the host signal by modifying either the DFT coefficients or the absolute values of the MDCT coefficients.

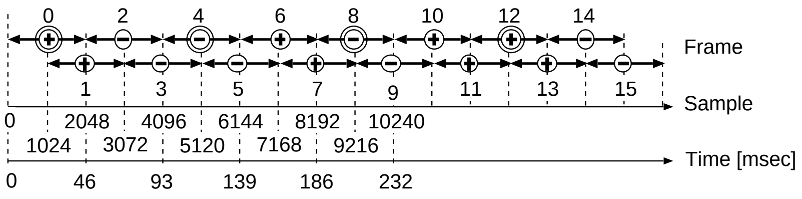

Figure 5.3: Frames used for embedding. The double-circled embedding frames are the beginnings of the tiles. The signs in the circles are examples of the signs used in a subband for each of the frames.

Table 5.1: Description of the abbreviations used in the figures. When a number is used with the abbreviations, it means the bitrate per a monaural channel used for AAC compression.

Abbreviation	Meanings
OrgPCM	Original unwatermarked uncompressed audio file
OrgAAC	Original unwatermarked AAC-compressed audio file
MagMdfyPCM	Magnitude modification in the uncompressed domain
MagMdfyAAC	Magnitude modification in the AAC-compressed domain
MagObsvPCM	Observation of the magnitude change in the uncompressed domain
MagObsvAAC	Observation of the magnitude change in the AAC-compressed domain
Trim	Trimming of the beginning part of the audio file. Various lengths of trimming are tested.
AACEnc	AAC encoding. A bitrate of 48, 64, or 96 kbps per a monaural channel is used.
AACDec	AAC decoding
WMEmbPCM	Watermark embedding in the uncompressed domain
WMDetPCM	Watermark detection in the uncompressed domain
WMEmbAAC	Watermark embedding in the AAC-compressed domain
WMDetAAC	Watermark detection in the AAC-compressed domain

(a) Basic concepts

The watermark embedding and detection algorithms for the uncompressed domain examined in this section are basically the same as the basic algorithms, only with different values of the parameters such as the length of the DFT frames, the watermark embedding strength, the length of the codeword, etc. The watermark embedding and detection algorithms for the compressed domain are modified versions of the basic algorithms for the uncompressed domain. We describe the algorithms below.

Overlapping frames We show Fig. 4.4 here again (Fig. 5.3) to explain the relationship with the frames and magnitude modification by the watermark embedding. A tile consists of four consecutive frames of either DFTs or MDCTs. In each four adjacent frames, the opposite signs are used for the first two frames and the last two frames of the four frames. Each of the frames overlaps the adjacent frames by a half window. In the figure, the double-circled embedding frames are the beginnings of the tiles. The signs in the circles are examples of the signs used in a subband for each of the frames.

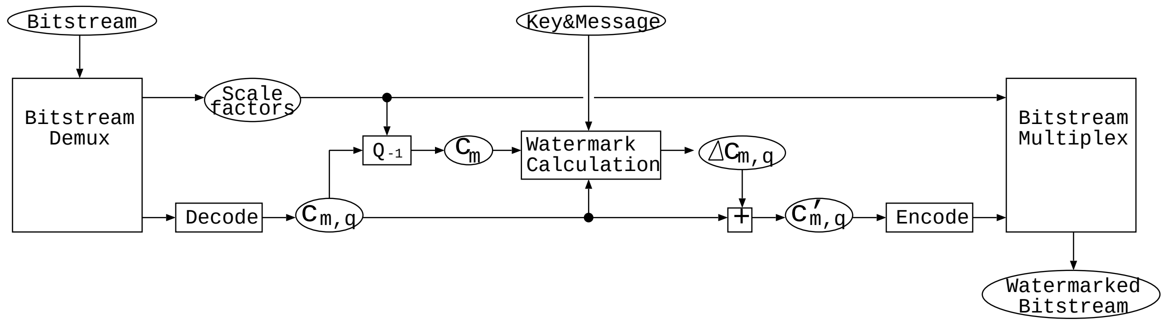

Figure 5.4: The process flow of the AAC watermark embedding algorithm.

Magnitude modification The embedding algorithm introduces non-uniformity of the magnitude distribution of the content in the frequency domain. When watermark embedding is performed for uncompressed content, the magnitudes of the complex DFT coefficients calculated by the short-term DFTs are increased or decreased according to the pseudo-random number. When embedding is performed for AAC-compressed content, the absolute values of the MDCT coefficients recovered from the bitstream are increased or decreased instead.

The detection algorithm calculates watermark detection strengths by correlating the magnitudes of the content and the pseudo-random numbers. While the watermark detection algorithm for uncompressed content uses the magnitudes of the complex DFT coefficients, the watermark detection algorithm for AAC-compressed content uses the absolute values of the recovered MDCT coefficients.

Watermark embedding in the AAC-compressed domain The watermark embedding algorithm for AAC-compressed content is illustrated in Fig. 5.4. The algorithm first decodes the quantized MDCT coefficients, $c_{m,q}[f,k]$, of the k-th frequency bin in the f-th frame of a pattern block of the content. The non-uniformities of the magnitudes are introduced by adding $\Delta c_{m,q}$ to $c_{m,q}$. In the last step, the modified quantized MDCT coefficients, $c_{m,q}{}'$, are encoded using the original Huffman codebooks and the original scale factors. Hence, requantization of the modified coefficients is not necessary. However, in order to determine how many coefficients are to be modified, an inverse quantizer, Q^{-1}, has to be applied to obtain the MDCT coefficients, c_m, before the watermark calculation process.

$$\begin{aligned} c_m[f,k] &= Q^{-1}(c_{m,q}[f,k]), && (5.1)\\ c_a[f,k] &= |c_m[f,k]|, && (5.2)\\ c_{m,q}{}'[f,k] &= c_{m,q}[f,k] + \Delta c_{m,q}[f,k]. && (5.3) \end{aligned}$$

The value of $\Delta c_{m,q}[f,k]$ is determined by the following simple rule, without using a psychoacoustic model. The total magnitude, $u[f,b]$, of the b-th subband in the f-th frame of a pattern block is calculated in each subband. The target value for the total magnitude after the modification is calculated as $(1+s_{tile}[t,b]\beta)u[f,b]$, while β is the degree of modification. The value of 0.1 was used in the experiments described below. From the lowest bin of the subband, whether or not $c_{m,q}[f,k]$ should be modified is examined. If modifying $c_{m,q}[f,k]$ makes the total magnitude of the subband closer to the target total magnitude, then it is modified. When there is no coefficient that satisfies the condition, or when the number of modified coefficients in the subband reaches a predefined limit, the modification calculation is finished. The limit numbers used for 48, 64, and 96 kbps-encoded monaural AAC content are 2, 4, and 8, respectively.

When increasing the magnitudes, the quantized coefficients are skipped if they already have the maximum absolute value of the codebook. Similarly, the zero coefficients are skipped when decreasing the magnitudes. The algorithm leaves the EIGHT_SHORT_SEQUENCE [30] frames unchanged, but modifies the LONG_START_SEQUENCE and LONG_STOP_SEQUENCE frames in the same way as the ONLY_LONG_SEQUENCE frames. Although utilizing a psychoacoustic model to control the watermark signal is desirable, it was not used in these experiments because it was difficult to adjust the quantized magnitudes according to the model's excessively detailed output.

Watermark detection in the AAC-compressed domain In the watermark detection algorithm in the AAC-compressed domain, after the quantized MDCT coefficients are decoded, an inverse quantizer is applied to them. The absolute values of the recovered MDCT coefficients are used for $a_{t,f}$. When there is an EIGHT_SHORT_SEQUENCE frame, the MDCT coefficients of the next frame are copied instead. The other types of sequences are handled in the same way. To allow for the calculation of logarithms, the value of 1 was given to the MDCT coefficients whose values were zero.

5.1.2 Magnitude modification experiments

In this section, it is shown that the magnitude changes made in either the uncompressed domain or the AAC-compressed domain can be observed in both of these . The abbreviations used in these figures are explained in Table 5.1.

In the test, magnitude modification is performed in either the uncompressed domain or the AAC-compressed domain. The magnitudes of the frequency coefficients are modified by the following simple rule without a key or a message. In the even subbands, the magnitudes are increased in the first two frames, decreased in the next two frames, and increased in the next two frames, and so on. In the odd subbands, the opposite signs are used. When observing the magnitude modification, the magnitudes of the modified content are compared to the content that is not modified. The logarithms of the magnitude summations of the even subbands are calculated, and the differences of the logarithms with and without magnitude modification are calculated. Then the mean of the differences is determined. In most cases, the frames used for the magnitude observation are displaced by trimming the beginning of the modified content[2].

(a) Magnitude modification in the uncompressed domain

In this test, the magnitudes of each of the frequency bins are modified by $\pm 10\%$ using short-term DFTs in the uncompressed domain (MagMdfyPCM in Fig. 5.5). The magnitude change is observed in the uncompressed domain just after trimming (E1a), in the compressed domain after AAC compression (E1c), or in the uncompressed domain after decompression of the AAC encoded content (E1b). More precisely, MagObsvAAC compares the logarithms of the summation of the absolute values of the recovered MDCT coefficients in the even subbands of the content that has been modified, trimmed, and compressed and those of the content that has been trimmed by the same length, and compressed.

The results are shown in Fig. 5.6. The horizontal axis of Fig. 5.6 is the number of trimming samples, that is, the displacement of the observation frames to the original modification frames. The vertical axis is the mean of the logarithmic magnitude difference. The graph (E1a) in Fig. 5.6 is the magnitude change observed in the uncompressed domain just after trimming,

[2]To observe the effect of magnitude modification in isolation, the unmodified content is also trimmed by the same length before the comparison.

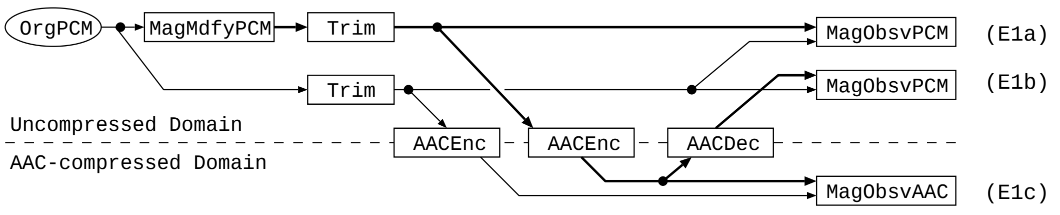

Figure 5.5: Settings of a preliminary experiment where the magnitude of a content is modified in the uncompressed domain and the magnitude change is observed in both of the compressed domain and the uncompressed domain.

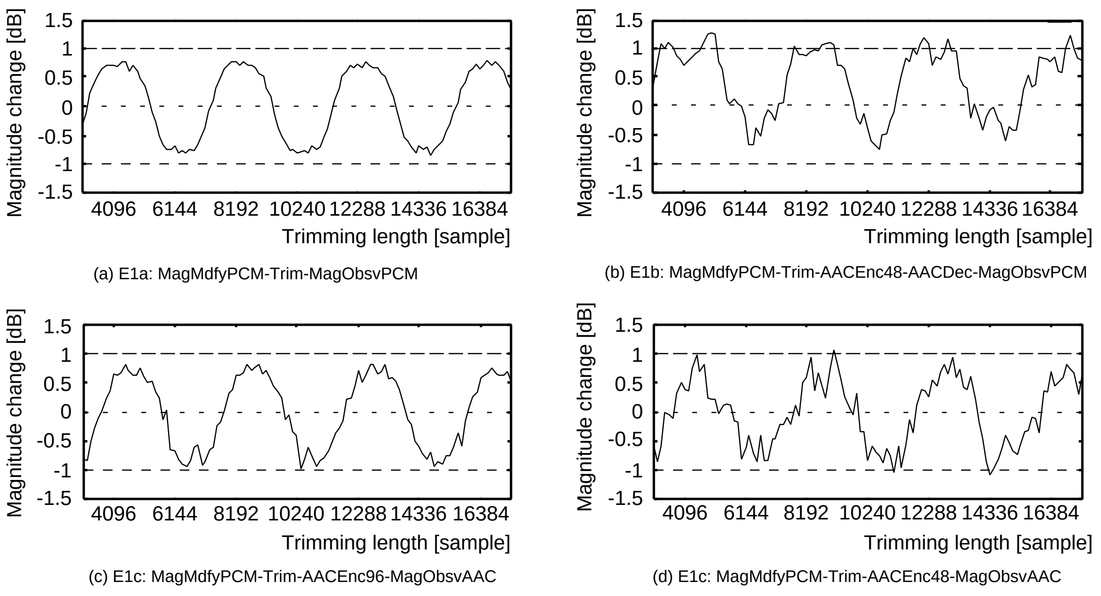

Figure 5.6: Experimental results of observation of magnitude change made in the uncompressed domain.

and it corresponds to the path (E1a) in Fig. 5.5. The graph (E1b) is the magnitude change observed in the uncompressed domain after AAC compression with a bitrate of 48 kbps per channel and decompression, and it corresponds to the path (E1b) in Fig. 5.5. The two graphs of (E1c), both of which correspond to (E1c) of Fig. 5.5, are observed in the compressed domain after compression with a bitrate of 96 and 48 kbps per channel, respectively. It can be seen that the magnitude change can be observed even in the observation frames in a different domain that are not exactly the same as the modification frames.

(b) Magnitude modification in the AAC-compressed domain

The total magnitudes of the subbands are modified by $\pm 10\%$ in the AAC-compressed domain (MagMdfyAAC in Fig. 5.7) using the rule described in Section (a). The results are shown in Fig. 5.8. The magnitude changes are observed in the compressed domain just after the modification (E2c), in the uncompressed domain after decompression (E2a), or in the compressed domain after re-compression (E2b). The magnitude modifications were done to an AAC file whose bit rate is 96 kbps per channel. The bitrates used for re-compression in the case of (E2b) were 96 and 48, respectively. It can be seen that magnitude modification is possible

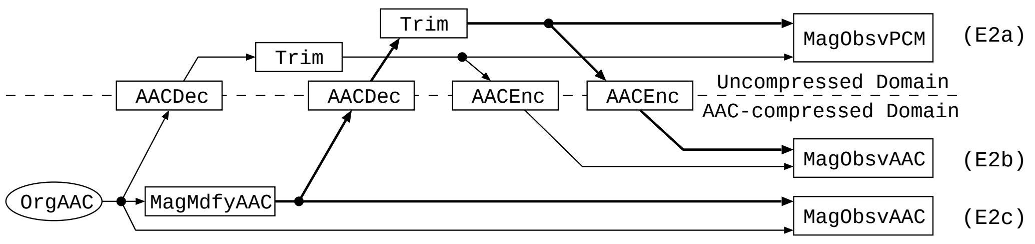

Figure 5.7: Settings of a preliminary experiment where the magnitude of a content is modified in the AAC-compressed domain and the magnitude change is observed in both of the compressed domain and the uncompressed domain.

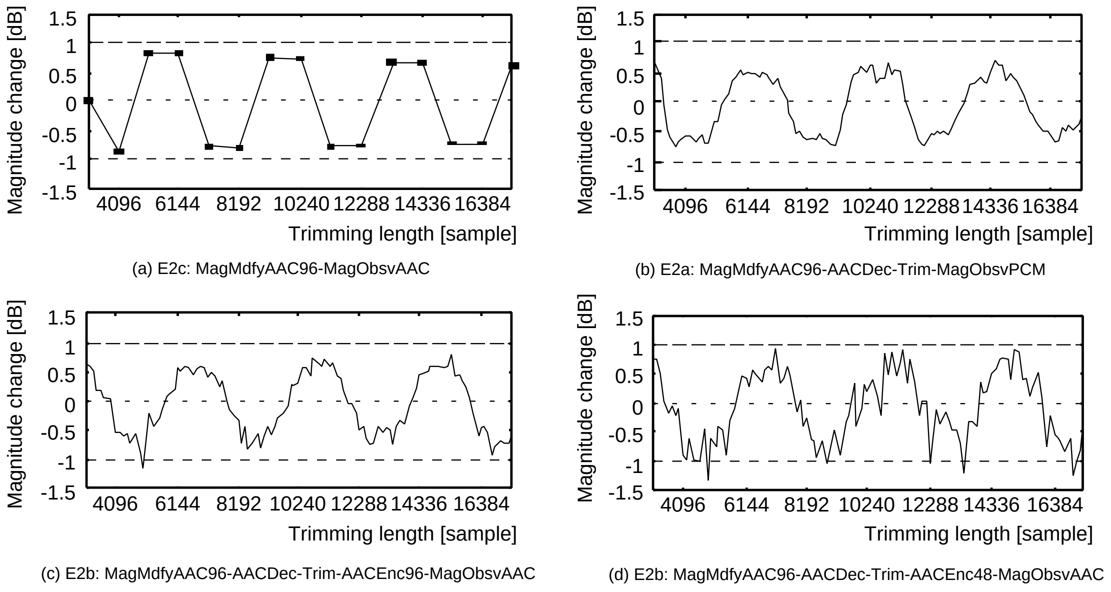

Figure 5.8: Experimental results of observation of magnitude change made in the AAC-compressed domain.

also in the compressed domain.

5.1.3 Results of robustness tests

In this section, the robustness of the watermarking method against AAC compression, decompression, and re-compression is shown. Nine pieces of monaural music such as pop music, orchestral music, and solo instruments were used for the robustness tests. All of the signals were sampled at a frequency of 44.1 kHz and with a bit resolution of 16 bits, and each piece was 100 seconds long.

(a) Implementation

We implemented methods that can embed 64-bit messages in 30-second pieces of music. The message is encoded in 448 bits by adding 8 Cyclic Redundancy Check (CRC) parity bits, using Turbo Coding, and repeating it twice. Each pattern block has 12 bits and a synchronization signal embedded, and the block has 28 columns and 8 rows of tiles. Each of the 28 frequency

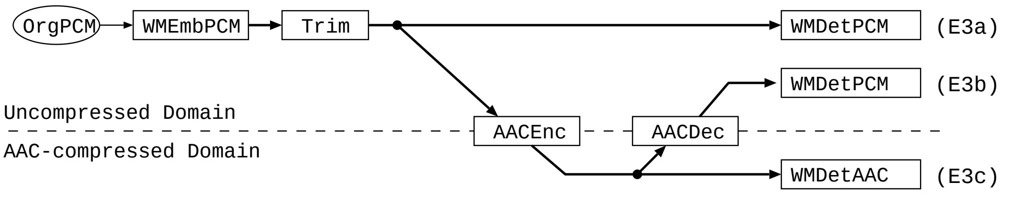

Figure 5.9: Settings of the robust test where watermark embedding was performed in the uncompressed domain.

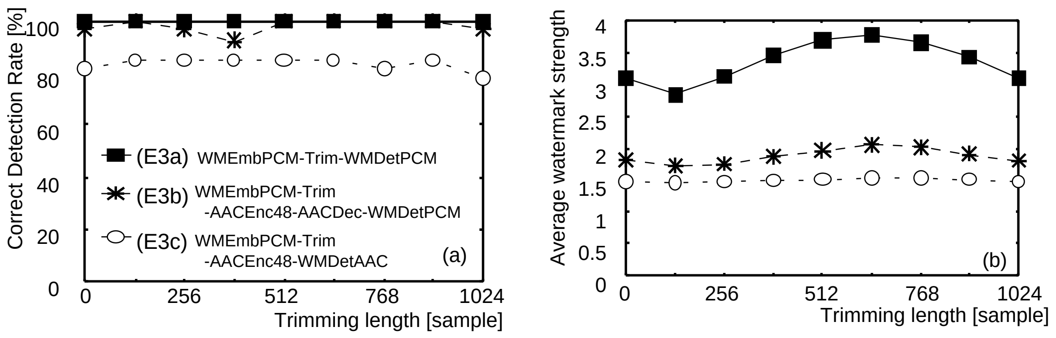

Figure 5.10: Robustness of watermark embedded in the uncompressed domain.

subbands is given an equal bandwidth of 20 frequency bins. The frequency of the highest bin used is 12.5 kHz. The length of a DFT frame is 2,048 samples, which is as long as the MDCT frames used for AAC compression. In the tests, no additional tools defined in the AAC specification [30] were used (such as temporal noise shaping (TNS) or prediction).

At the time of detection, while 32 tiles out of the 224 tiles are dedicated for the local adjustment of the pattern block synchronization, the tiles assigned for the bits are also used for the global synchronization. For the global synchronization, it was assumed that 7 consecutive blocks would always have consistent synchronization positions. The false alarm error ratio is theoretically under 10^{-5} based on the threshold of the square means of the detected bit strengths. Another threshold on the estimated watermark signal-to-noise ratio (SNR) is set to keep the code word error ratio under 10^{-5}. The reasons to use both thresholds are described in Ref.[61]

We measured the correct detection rates (CDRs) and the mean of the watermark strengths, X, at which the correct 64-bit messages were detected. The error correction and detection algorithm successfully avoided detection of an incorrect message.

(b) Robustness of watermark embedded in the uncompressed domain

The watermark was embedded in the uncompressed domain (WMEmbPCM in Fig. 5.9). The root mean square power of the watermark signal was on average 23.9 dB lower than that of the host signal. Watermark detection was done in the uncompressed domain just after trimming (E3a), in the compressed domain after AAC compression (E3c), or in the uncompressed domain after decompression of the AAC encoded content (E3b).

The experimental results are shown in Fig. 5.10. The horizontal axis of the graphs is the number of trimming samples. Because the synchronization process chooses the best frame in the detection frames, and that the interval between a frame and the next frame is 1,024 samples, the trimming of more than 1,024 samples was not tested. While the vertical axis of the left graph is the CDRs, that of the right graph is the average watermark strength. A

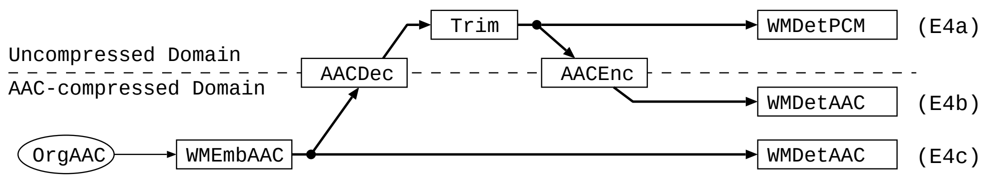

Figure 5.11: Settings of the robustness experiments where watermark embedding was performed in the AAC-compressed domain.

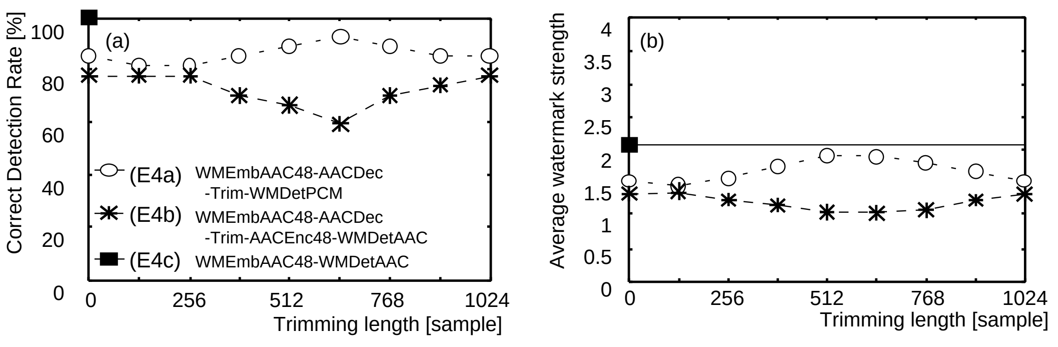

Figure 5.12: Robustness of watermark embedded in the AAC-compressed domain.

bitrate of 48 kbps per channel was used for the compression. CDRs over 80% were seen for every tested trimming length and every tested path.

(c) Robustness of watermark embedded in the AAC-compressed domain

Tthe watermark was embedded in the AAC-compressed domain (WMEmbAAC in Fig. 5.11). The root mean square powers of the watermark signal for 48, 64, and 96 kbps per channel AAC audio were on average 19.3, 22.0, and 26.6 dB lower than that of the host signal, respectively. The SNR of AAC compression by the AAC encoder used for the experiment was 18.1, 23.4, and 29.9 dB for these bitrates, respectively. Watermark detection was done in the compressed domain just after embedding (E4c), in the uncompressed domain after decompression (E4a), and in the compressed domain after re-compression (E4b).

The experimental results are shown in Figures 5.12 to 5.14. In Fig. 5.12, while watermark embedding was done for AAC files compressed with a bitrate of 48 kbps per channel, detection was done in the uncompressed domain (E4a) and the uncompressed domain (E4b) and (E4c). Because trimming finer than 1,024 samples is impossible for AAC compressed files, the result of (E4c) is constant. A bitrate of 48 kbps per channel was again used for re-compression of the decoded audio files for (E4b) of the figure. CDRs for detection in the re-compressed audio files were 60-80%. The reason the peaks of the CDRs are at different trimming lengths is not yet clear. It is possibly because the AAC encoder might insert some samples at the beginning of the audio file to be encoded in order to preserve audio quality.

Fig. 5.13 and Fig. 5.14 show the results of compressed-domain detection after re-compression for various cases. The original compression bitrates for Fig. 5.13 and 5.14 were 96 and 48 kbps, respectively, while detection was performed at the bitrates of 48, 64, and 96 kbps. For the 96 kbps-compressed AAC embedding, the effect of the detection bitrate can be seen more prominently.

We showed the basic algorithm described in Section 4 can be used as watermarking methods

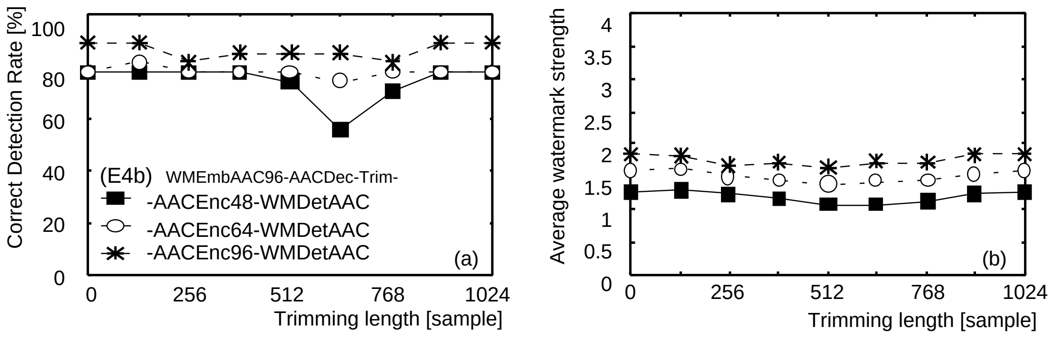

Figure 5.13: Detection of watermark embedded in AAC files with a bitrate of 96 kbps per channel after the files are decompressed, trimmed and compressed again (E4b of Fig. 5.12). 48, 64, or 96 kbps was used for re-compression.

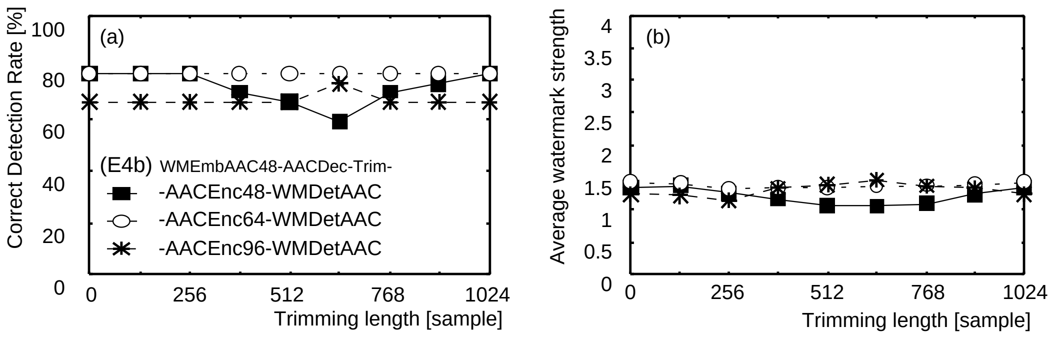

Figure 5.14: Detection of watermark embedded in AAC files with a bitrate of 48 kbps per channel after the files are decompressed, trimmed and compressed again (E4b of Fig. 5.12). 48, 64, or 96 kbps was used for re-compression.

for the AAC-compressed domain and the uncompressed domain. Experimental results showed that the watermark embedded in an uncompressed audio file was successfully detected in the AAC-compressed domain after the file is compressed and that the watermark embedded in an AAC file was able to be detected after the file was decompressed, trimmed, and compressed again.

5.2 Audio Watermarking for Live Performance

Until this point, we have been dealing with audio watermarking as a tool for copy control of digital music. However, audio watermarking is not a technology useful only for digitally stored music. Music is performed, created, stored, and listened to in many different ways, and it is much more common that music is not stored as a digital file on a computer. When applying audio watermarking technology in various musical environments, *real-time watermark embedding* is a preferred approach. For example, in music mastering studios, by embedding a watermark in real time in a sound being played, the watermarked sound could be instantly checked without saving it as a file on a computer. For broadcasting of a watermarked sound, it would become possible to embed a watermark in real time and instantly broadcast the sound of a live performance being performed in a studio, or the voice of a newscaster or a talk show.

It is not difficult to perform watermark embedding faster than playback using current

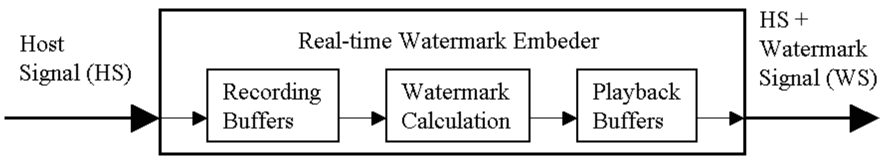

Figure 5.15: C1: Naive Composition Method: A mixture of a host signal (HS) and a watermark signal (WS) is calculated for a short HS stored in a recording buffer, and then is played back from a playback buffer.

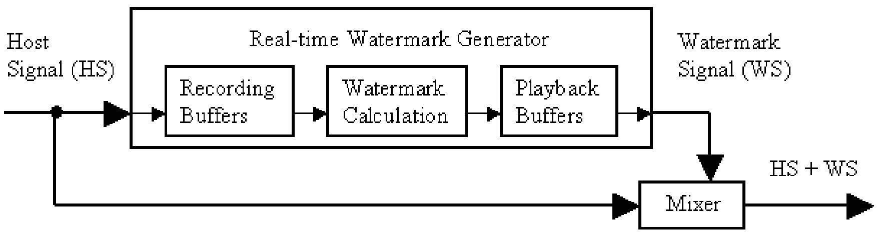

Figure 5.16: C2: Analogue Watermarking: The computer outputs only the WS. Mixing is performed at a trusted conventional analogue mixer outside of the computer. Control of the volume of the WS is much easier.

audio watermarking technologies and computers. However, there are other problems besides the known problems for file embedding. In this section, we classify several composition methods for real-time watermark embedding, and point out problems that occur with the composition methods, discuss a real-time embedding method, and report the results of some experiments.

5.2.1 Real-time watermark embedding for live performance

In this section, we describe several system compositions for real-time embedding, and point out their merits and flaws.

(a) C1: Naive Composition

Most watermarking algorithms are designed to be given a host signal (HS) and to produce a signal that is mixture of the HS and a watermark signal (WS). The most naive composition method by simply applying an algorithm of this kind to real-time embedding is illustrated in Fig. 5.15. In this composition method, a HS with a certain length is stored in a recording buffer. The stored HS is given to the watermark embedding process where a mixture of the HS and a WS is calculated. The mixture is then stored in a playback buffer and played at the proper time. Any audio watermarking algorithm can be used for real-time embedding with this composition method as long as it runs faster than the playback.

However, there are two drawbacks with this composition method. The first drawback is the inevitable delay of the HS (and the WS). The recording buffer and the playback buffer must have some length so that the mixed signal can be played steadily. If the buffers are longer, stability of playback increases at the expense of the delay of the HS. The second drawback is risk of interrupting of the playback. When a problem blocks the real-time watermark embedder, playback of not only the WS but the HS stops. This would make a big problem for a live broadcast of a watermarked signal.

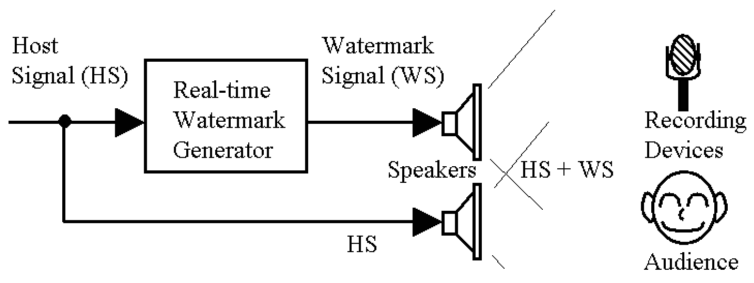

Figure 5.17: C3: Sonic Watermarking: The WS sound and the HS sound enter the air from separate speakers. Both sounds mix in the air and form a watermarked sound so that the embedded message could be detected from a recorded sound.

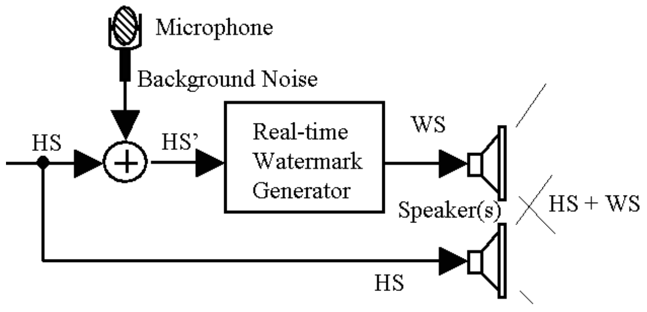

Figure 5.18: C4a: Noise Recording: The background noise of the environment is recorded using a microphone and mixed with the host signal so that the frequency masking effect of the noise can be also taken into account.

(b) C2: Analogue Watermarking

To solve the problem of C1, another composition method illustrated in Fig. 5.16 mixes the HS and the WS outside the computer using a trusted conventional analogue mixer after the WS is converted to an analogue signal. The computer, a real-time watermark generator, should be designed to output only the watermark signal. The reason that the computer needs to be fed the HS is that it utilizes the HS for calculating the frequency masking effect[84] of the HS. This composition does not introduce additional delay for the HS. If a problem affects the computer, while generation of the WS would stop, the broadcast of the HS is still secure. In addition, it is very easy for a sound engineer to control the volume of a WS using a familiar mixer. This would make it easy for sound engineers to use a watermark embedder.

That the WS is delayed relative to the HS because of the buffers and the watermark calculation can be a problem of this composition method.

(c) C3: Sonic Watermarking

Because both the HS and the WS are sounds, it is possible to mix them in the air (Fig. 5.17). In this composition method, a real-time watermark generator the same as the one used in C2 outputs the WS, and the WS enters the air from a speaker. The WS sound and the HS sound mix in the air and make a watermarked sound.

This composition method will be useful for live concerts in which electronic instruments are used. In concerts of popular music, because the sounds of the instruments and the vocals

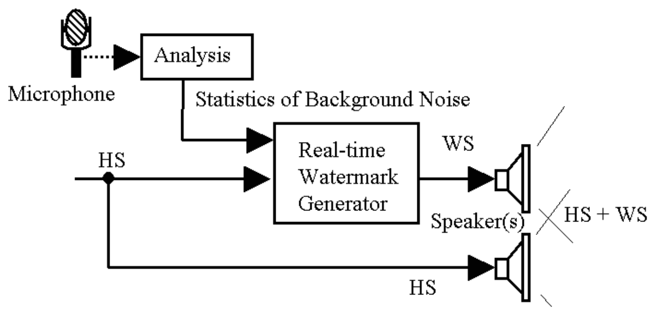

Figure 5.19: C4b: Noise Prediction: If the background noise is known to be stationary, its frequency masking effect can be predicted without recording during the performance.

are electronically captured and amplified, it would be easy to snatch the sound signal and feed it to a watermark generator just before amplification for the speakers. Using watermark embedding by this composition method, it would be possible to find *bootleg recordings* on the Internet, that is, illegal music files that have been recorded in concert halls by unethical audience members using recording devices.

This composition method also has the same potential problem of the delay of the WS relative to the HS. Another problem of this composition method is that background noises made by sources besides the musical instruments become disturbing factors for watermark detection. Such sounds include voices and clapping by audience members, reverberations of the hall, and rustling noises made by hands touching the recording device.

(d) C4: Sonic Watermarking with Sampling

A method to make a watermark detectable even when there is a loud background noise is to also make use of the frequency masking effect of the noise. Figure 5.18 illustrates a composition method, C4a, where background noises are recorded using a microphone, added to the host signal played by musical instruments, and finally input to a real-time watermark generator. When there is a loud background noise, the WS also increases, and hence detection of the embedded message would be easier. However, there may be several problems in the recording of background noises. It is impossible to record noises that are made near scattered recording devices. Recoding only the background noise while excluding the sound of the musical instruments is also difficult.

If the background noise of the environment is known to be stationary, recording of the noise during the performance is not necessary. The watermark generator can predict the frequency masking effect of the noise as long as its power spectrums are analyzed prior to the performance (Fig. 5.19). Because this composition method, C4b, does not follow the dynamic changes of the noise, the successful rate of detection of embedded message will be lower than for C4a. For both C4a and C4b, the actual masking effects of each audience member depend on their locations. Arrangements using multiple speakers will be better. When using multiple speakers, it is also necessary to consider their mutual interference.

A composition method, C4c, illustrated in Fig. 5.20 is the only composition method that can be applied to musical instruments that do not use electronic amplification. The only difference in C4a is that sounds of the instruments together with the background noises are also recorded by a microphone for the watermark calculation. For mixing in the air, we believe that it does not matter whether the HS sound is electronically amplified or acoustic. This composition method can dynamically respond to changing background noises.

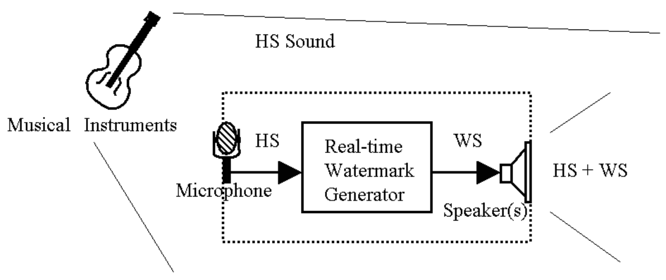

Figure 5.20: C4c: Host Signal Recording: When the musical instruments do not use electronic amplification, their sound together with the background noises should be recorded by a microphone for watermark calculation.

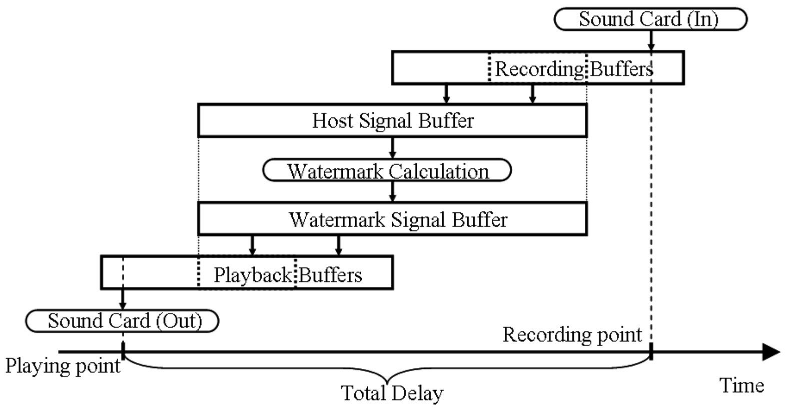

Figure 5.21: A watermark signal (WS) is delayed relative to a host signal (HS) because of recording buffers, watermark calculation, and playback buffers.

5.2.2 Problems of real-time watermark embedding

As long as the composition method C1 is chosen for real-time embedding, there is no problem in applying the previous method. In that case, nothing is different from handling files stored on a computer. However, applying to compositions methods C2 to C4 results in an inevitable delay of the watermark signal (WS) relative to the host signal (HS) which causes a serious problem.

The delay is caused by the recording buffers, playback buffers, and WS calculation (Fig. 5.21). Although the length of the playback buffers and the recording buffers can be reduced using ASIO[3] software and hardware, it is impossible to reduce them to zero. A total of 128 samples for each buffer is required for stable real-time embedding for our experiment[4]. The WS calculation causes two kinds of delay. The first is that it is necessary to store one Discrete Fourier Transform (DFT) frame of the HS to calculate its power spectrums. The second is the elapsed time for the WS calculation. The watermark calculation takes approximately 10% of the playback time in our experiment. For example, if the length of a DFT frame is 512 samples, the elapsed time for the WS calculation corresponds approximately to playback time for 51.2

[3]ASIO is the Steinberg Audio Stream Input/Output architecture for low latency high performance audio handling.

[4]We used a Mobile Pentium III 1.13 GHz notebook computer with a docking station equipped with a LynxOne sound card by Lynx Studio Technology.

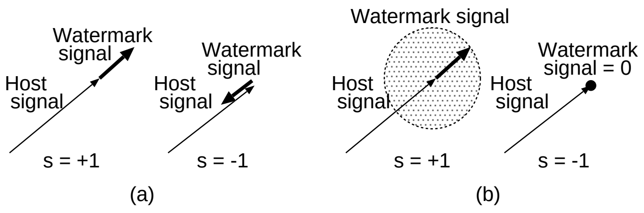

Figure 5.22: The host signal and the watermark signal for (a) the previous method and for (b) the proposed method.

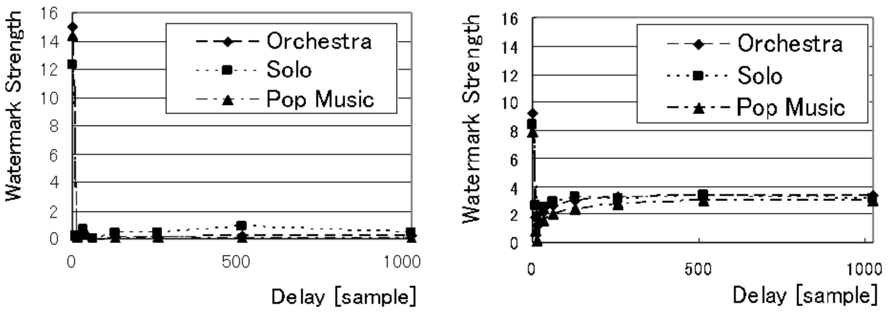

Figure 5.23: Mean of detected watermark strengths from mixtures of a watermark signal (WS) and a host signal (HS) for the previous method (a) and the new method (b). The WS is delayed by N_d samples relative to the HS. A delay as small as eight samples ruins detection.

samples. Hence, using 512-sample DFT frames, the total delay is $128+128+512+51.2 = 819.2$ samples, which is about 18.5 ms for 44.1 kHz sampling.

This delay makes perfect synchronization of the WS and the HS impossible. The embedding algorithm of the previous method embeds the watermark by modifying amplitudes in the frequency domain while preserving the phases (Fig. 5.22a). When the watermark signal is delayed by at least 18.5 ms, since the phase of the HS drastically changes during the delay, the phases of the HS and the WS become almost independent. Accordingly, decreasing the amplitudes in the tiles where negative pseudo-random numbers are assigned becomes impossible.

Figure 5.23(a) is the result of an experiment where we calculated a WS for a HS using the previous method, delayed the WS by N_d samples, overlapped it with the HS, and observed the watermark strengths detected from the mixture. The horizontal axis of the figure is the given delay, N_d. The vertical axis is the mean of detected watermark strengths. The distribution of the strengths for unwatermarked content can be regarded as a normal distribution whose variance is 1.14. The reason the variance for unwatermarked content is not unity is that, since the detected strengths for bits are also used for the synchronization, the positions with large strengths are selected in the searching process. It can be seen that as small a delay as eight samples (approximately 0.02 ms) made detection impossible. Consequently, the previous algorithm cannot be used for compositions methods C2 to C4.

5.2.3 Modified algorithms for real-time watermark embedding

We altered the embedding algorithm describe in Section 4 as described below so that modification of amplitudes is possible under the situation where the watermark signal is delayed relative to the host signal.

The only change is to make the values of the magnitudes $z_a[f, k]$ of the WS zero for the tiles with a negative sign (Fig. 5.22b2). This is because it is impossible to decrease the magnitudes of the HS. As for the tiles with a positive sign, the magnitudes and the phases of the WS are given as in the previous method. However, because of the delay, to give the WS the same phases as the HS has almost the same effect as giving the WS a random phase (Fig. 5.22b1).

This change makes the power distribution of the content non-uniform, and hence makes detection possible. However, because the efficiency of amplitude modification is much worse than in the previous algorithm, a decrease of the detected watermark strength is inevitable. It is necessary to use a stronger watermark signal than the previous method uses.

Note that the alteration is only for embedding, and that the same previous detection algorithm can detect the watermark from the content whether the previous algorithm or the altered algorithm is used for watermarking.

5.2.4 Experimental results

In this section, we report results of experiments applying the new method to the real-time embedding composition methods C2 to C4.

(a) Implementation

We implemented the method in a software system that can embed and detect 64-bit messages in 30-second pieces of music. The message is encoded in 448 bits by adding 8 Cyclic Redundancy Check (CRC) parity bits, using Turbo Coding, and repeating it twice. Each pattern block has 3 bits and a synchronization signal embedded, and the block has 24 columns and 8 rows of tiles. Each of the 24 frequency subbands is given an equal bandwidth of 6 frequency bins. The frequency of the highest bin used is 12.7 kHz. While 48 tiles out of the 192 tiles are dedicated for the local adjustment of the pattern block synchronization, the tiles assigned for bits are also used for the global synchronization. For the global synchronization, It is assumed that 16 consecutive blocks have consistent synchronization positions. The length of a DFT frame is 512 samples to shorten the delay. The false alarm error ratio is theoretically controlled to be under 10^{-5} by the threshold of the square means of the detected bit strengths. Another threshold on the estimated watermark signal-to-noise ratio (SNR) is set to keep the code word error ratio under 10^{-5}. The reason to use double thresholds is described in Ref.[61]. Nine pieces of monaural music from pop music, orchestral music, and solo instruments were used for the analysis. All of the signals were sampled at a frequency of 44.1 kHz and with a bit resolution of 16 bits, and each piece is 100 seconds long.

(b) Psychoacoustic model

The ISO-MPEG 1 Audio Psychoacoustic Model 2 for Layer III [29] is used as the basis of the psychoacoustic calculation for the experiments with some alterations.

- An absolute threshold was not used for these experiments. We believe this is not suitable for practical watermarking because it originally depends on the listening volume and is too small in the frequencies used for watermarking.

- A local minimum of masking values within each frequency subband was used for all frequency bins in the subband. Not only excessively changing the WS magnitudes does not contribute to the watermark strength but it causes lower acoustic quality by increasing the WS without effect.

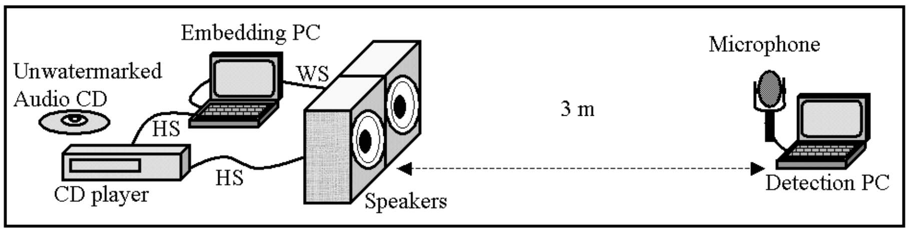

Figure 5.24: An experimental system for testing C3 and C4b was set up in a soundproof room where a 30 dB(A) background noise was heared when nothing was played by the speakers.

- A 512-sample frame, 256-sample IBLEN, and a sine window were used for the DFT for the psychoacoustic analysis to reduce the computational cost.

It is known that postmasking stays as high as simultaneous masking for a 5 ms delay and that it starts an almost exponential decay with a time constant of 10 ms after the first 5 ms [84]. Hence, a shorter DFT frame is expected to result in better acoustic quality because of the shorter delay. However, the poor frequency resolution caused by a too short DFT frame reduces the detected watermark strength. This is the reason a 512-sample DFT frame was selected for the implementation.

Based on the psychoacoustic model, the root mean square power of the WS is 20.4 dB lower than that of the HS on average. Although we have not conducted systematic testing of the acoustic quality of the method, we noticed that the watermarked songs tend to sound a little hoarse. We think that one reason for the hoarseness is the fast switching of the WS in short DFT frames.

(c) Experiment of Analog Watermarking (C2)

Figure 5.23(b) is the result of the same experiment as in Fig. 5.23(a) using the new method. It can be seen that the detected watermark strength of this method with a delay of the WS is greater than in the previous method.

A test of analogue watermarking was conducted. The WS was mixed with the HS with a delay of around 830 samples by using an analogue mixer. The mixed analogue signal was converted to a digital signal using a sound card in a notebook personal computer (PC) to detect the watermark. The *Analogue Watermarking* row of Table 5.2 shows the mean of the detected watermark strengths and the correct detection rate for the message.

(d) Experiment of Sonic Watermarking (C3)

The system illustrated in Fig. 5.24 was set up in a soundproof room for experiments with sonic watermarking. Although the room was soundproofed, a 30 dB(A)[5] background noise was observed when nothing was played by the speakers. Two speakers with a built-in amplifier were placed next to each other to play the HS and the WS, respectively. The playback volumes of the speakers were set at the levels playing the same signal with the same sound level. The mixed sound of the HS and the WS was captured by a microphone and sent to the sound card of the notebook PC where the signal was converted to a digital signal to detect the watermark. The distance between the speakers and the microphone was 3 m. When a loud popular music song was played using the speakers, the sound level was from 70 dB(A) to 75 dB(A) at the position of the microphone.

[5]dB(A) is a unit for the A-weighted sound level[16].

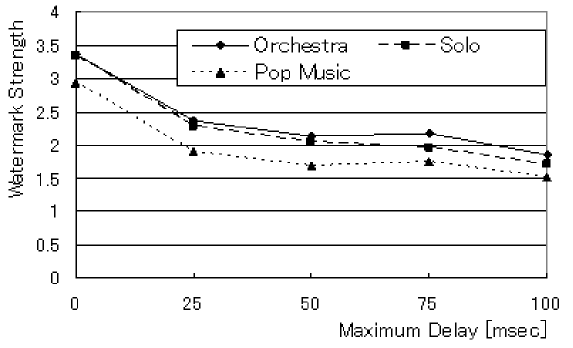

Figure 5.25: Mean of detected watermark strengths after echo addition. The horizontal axis is the value of a maximum delay. Neither sonic watermarking nor analogue conversion was not used.

The results of the experiment are shown in the *Sonic Watermarking* column of Table 5.2. The reason the detected strengths for solos and orchestras are too small to stably detect the messages is because the solo instrumental music and orchestral music have wider dynamic ranges than popular music does, and they contain more low volume portions. In particular, a solo performance of a piano has most of its power in its low frequencies, but almost no power in its middle and high frequencies. Because of the existence of the 30 dB(A) virtually white background noise, the host signal to noise ratio in those portions became too low to detect the watermark. This is the same problem described in Section (c) about background noises including voices and applause from the audience.

It is likely that the echo of the room will cause a detection problem in practical situations. The means of watermark strengths detected after echo addition are shown in Fig. 5.25. The horizontal axis is the value of a maximum delay used for echo addition. The value of 0.5 was used for the feedback coefficient for echo addition. Although the strengths were greatly decreased, correct messages were detected from every one of the tested degradations. Neither sonic watermarking nor analogue conversion was not used for this test.

(e) Experiment of Microphone Recording (C4)

Composition methods C4a to C4c are expected to be more robust in an environment with background noises. To implement these composition methods, a calibration step is necessary between an amplitude digitally represented in the software and the corresponding actual sound played from the speakers. Without the calibration, it is impossible to generate the WS to be masked by the frequency masking effect of the sounds in the air. Calibration can be done by playing a known reference sound containing several pure tones at various frequencies, recording on the computer with a microphone the sound coming from the speakers, and calculating the ratio of the amplitudes of the played sounds to those of the recorded sounds.

We selected the composition method C4b to investigate the effect of utilizing the masking effect of the background noise, because the background noise of our experimental room can be considered to be stationary.

The background noise when no music was being played was recorded by the microphone. The power spectrums were calculated using a DFT and stored on the computer after multiplying by the calibration ratios. When real-time watermark generation is in progress, the stored power spectrums of the background noise are added to the power spectrums of the host signal for calculation of the frequency masking. The result of the experiment is shown in the

Table 5.2: The means of detected watermark strengths (Mean) and the rates (CDR) by which the correct 64-bit message was detected.

	Pop Music		Orchestra		Solo	
	Mean	CDR	Mean	CDR	Mean	CDR
File Watermarking	2.93	100.0%	3.37	100.0%	3.34	100.0%
Analogue Watermarking	3.82	100.0%	3.09	100.0%	2.51	100.0%
Sonic Watermarking	3.49	100.0%	1.46	66.7%	1.21	66.7%
Sonic Watermarking with Sampling	3.56	100.0%	1.97	100.0%	1.80	100.0%

Sonic Watermarking with Sampling row of Table 5.2. It can be seen that the correct detection rates for the solos and the orchestras were improved. When nothing or quiet music was being played, we could hear the sound of the WS coming out of the speaker. However, the reason we could hear the noise was mainly because we could localize the source of the noise at the speaker, and the noise itself was not unpleasant.

In this section we described several composition methods for real-time audio watermark embedding, one of which was *sonic watermarking* to mix the host signal and the watermark signal in the air. To solve a problem found in applying the basic algorithm to the composition, we slightly altered the embedding algorithm. Because the change is only for embedding, the same detection algorithm can be used for content watermarked by the previous method and content watermarked by the new method. The successful results of the experiments showed the possibility of the compositions methods. Since *sonic watermarking* is particularily a new and challenging application of audio watermarking, we further discuss the application scenarios in detail and report results of additional experiments in the next section.

5.3 Sonic Watermarking

We introduced *Sonic Watermarking* in the previous section. This composition method mixes the sound of the watermark signal (WS) and the host sound in the air so that the watermark can be detected from a recording of the mixed sound. The method will allow searching for *bootleg recordings* on the Internet, that is, illegal music files that have been recorded in auditoriums by unethical audience members using portable recording devices. The recordings are sometimes burned on audio Compact Discs (CDs) and even sold at shops, or distributed via the Internet. Countermeasures such as examining the audience members' personal belongings at auditorium entrances have been used for decades to cope with this problem. The ease of distribution in the broadband Internet has increased the problem of bootleg recordings.

In this section, we carefully consider the application model and the possible problems of sonic watermarking and report the results of intensive robustness tests and a MUSHRA[6] subjective listening test which we performed to investigate the effects of critical factors of sonic watermarking, such as the delay and the distance between the sound sources of the host signal (HS) and the WS. In Section 5.3.1, we describe the usage scenario of sonic watermarking. Some possible problems limiting use of sonic watermarking are listed in Section 5.3.2. In Section 5.3.3, we describe a watermarking algorithm that is designed to solve some of the problems. The robustness of the algorithm is shown by experimental results in Section 5.3.4. The acoustic quality of the algorithm is assessed by a subjective listening test described in Section 5.3.5.

[6]MUltiple Stimulus with Hidden Reference and Anchors [31].

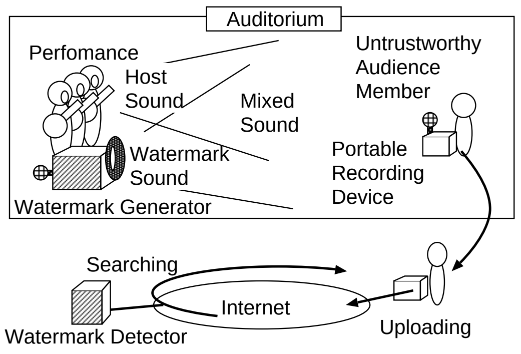

Figure 5.26: Sonic watermarking to detect bootleg recordings on the Internet. The watermark sound and the host sound are mixed in the air.

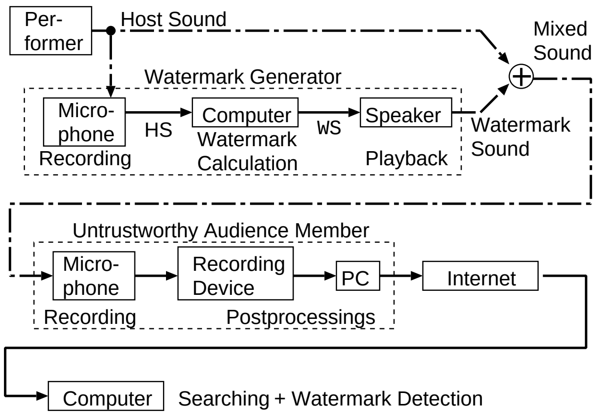

Figure 5.27: The lifecycle of a bootleg recording with sonic watermarks. While broken lines with arrowheads indicate sonic propagation, solid lines indicate wired analog transmissions or digital file transfers.

5.3.1 Usage scenario of sonic watermarking

In sonic watermarking, the watermark sound generated by a watermark generator is mixed with the host sound in the air (Fig. 5.3.1). A watermark generator is a device that is equipped with a microphone, a speaker, and a computer. The host sound is captured using the microphone, the computer calculates the WS, and the WS enters the air from the speaker. The reason that the computer needs to be fed the host sound is to calculate the frequency masking effect [84] of the host sound. The lifecycle of a bootleg recording containing sonic watermarks is illustrated in Fig. 5.27. While broken lines with arrowheads indicate sonic propagation, the solid lines indicate wired analog transmissions or digital file transfers. For example, the unethical audience member may compress the bootleg recording as an MPEG1 Audio Layer 3 (MP3[29]) file and upload it to the Internet. They may attack the sonic watermarking before compression. The recording device may be an analog cassette tape recorder, a MP3 recorder, a Mini-Disc recorder, etc.

Note that sonic watermarking is not necessary in live performances where the sound of the musical instruments and the performers are mixed and amplified using analog electronic devices. Analog watermarking [66] can be used instead.

5.3.2 Problems of sonic watermarking

In this section, we classify the possible problems that may limit the use of sonic watermarking into three major categories: (1) real-time embedding, (2) robustness, and (3) acoustic quality. Although all of the other problems of digital audio watermarking are also problems of sonic watermarking, they are not listed here.

(a) Problems related to real-time embedding

The major problems related to real-time embedding are the performance of the watermark embedding process and the delay of the WS.

1. Performance
   Watermark embedding faster than real-time is the minimum condition for sonic watermarking. The computational load of the watermark generator must be kept low enough for stable real-time production of the WS. A watermark embedding algorithm faster than real-time was also reported by Ref. [53].

2. Delay
   Even when the watermark generator works in real-time, the watermark sound will be delayed relative to the host sound. We will discuss the problems of robustness and acoustic quality caused by the delay in later sections.

   The delay consists of a pre-recording delay and a delay inside the watermark generator. The pre-recording delay is the time required for the sound to propagate from the source of the host sound to the microphone of the watermark generator. For example, when the distance is 5 m, the pre-recording delay will be approximately 15 ms.

   The delay inside the watermark generator is caused by the recording buffers, playback buffers, and WS calculations (Fig. 5.21). Although the length of the playback buffers and the recording buffers can be reduced using technologies such as ASIO[7] software and hardware, it is impossible to reduce them to zero. The WS calculation causes two kinds of delay. The first is that it is necessary to store a Discrete Fourier Transform (DFT) frame of the HS to calculate its power spectrums. The second is the elapsed time for the WS calculation.

(b) Robustness

Possible causes interfering with successful detection can be roughly categorized into (1) deteriorations after recording, and (2) deteriorations before and during recording by the unethical audience member. After recording, the unethical audience member may try to delete the watermark from the bootleg recording. The possible attacks include compression, analog conversion, trimming, pitch shifting, random sample cropping, etc. As for deteriorations before and during recording, the following items have to be considered:

1. Delay of the watermark signal
   When the WS is delayed the phase of the HS drastically changes during the delay, so the phases of the HS and the WS become almost independent. Watermarking algorithms assuming perfect synchronization of the phases suffer serious damage from the delay.

[7]ASIO is the Steinberg Audio Stream Input/Output architecture for low latency high performance audio handling.

2. Reverberations
   Reverberations of the auditorium must be mixed into the host sound and the watermark sound.

3. Noises made by audience
   Noises made by sources other than the musical instruments become disturbing factors for watermark detection. Such sounds include voices and applause from audience members, and rustling noises made by hands touching the recording device. If microphones directed towards the audience record the loud noise of the audience, and if the watermark generator utilizes the masking effect of the audience noise as well, detection of the watermark will be easier. However, since it is impossible to record noises that are made near widely scattered portable recording devices, the noise inevitably interferes with watermark detection.

4. Multiple watermark generators
   In some cases, arrangements using multiple watermark generators would be better to reflect the actual masking effects of each audience member. When using multiple watermark generators, it would be also necessary to consider their mutual interference.

(c) Acoustic quality

There are several factors that may make the acoustic quality of sonic watermarking worse than that of digital audio watermarking.

1. Strength of the watermark signal
   Because the efficiency of watermark embedding is worse and more severe deterioration is expected in the sound than for digital audio watermarking, the WS must be relatively louder than a digital audio watermark. This results in lower acoustic quality.

2. Delay of the watermark signal
   An example would be when the host sound includes a drumbeat that abruptly diminishes, and the delayed watermark sound stands out from the host sound and results in worse acoustic quality. There is a *postmasking effect* that occurs after the masker diminishes [84]. For the first 5 ms after the masker diminishes, the amount of the postmasking effect is as high as simultaneous masking. After the 5 ms it starts an almost exponential decay with a time constant of 10 ms. Therefore, if the delay of the watermark sound is short enough, the postmasking effect is expected to mask the watermark sound. However, the longer the delay, the more the host sound changes, and the weaker the masking from the postmasking effect.

3. Differences of the masker
   The HS captured by the microphone of the watermark generator is different from the host sound that the audience listens to. Hence, the masking effect calculated by the generator will also be different from the actual masking effect as heard by the audience.

4. Different locations of the sound sources
   While the sources of the host sound may be spread around the auditorium stage, the sources of the watermark sound must be limited to a few locations, even if multiple watermark generators are used. The difference in the direction and the distance of the sources of the watermark sound and the host sound from each audience member will have a negative effect on the acoustic quality.

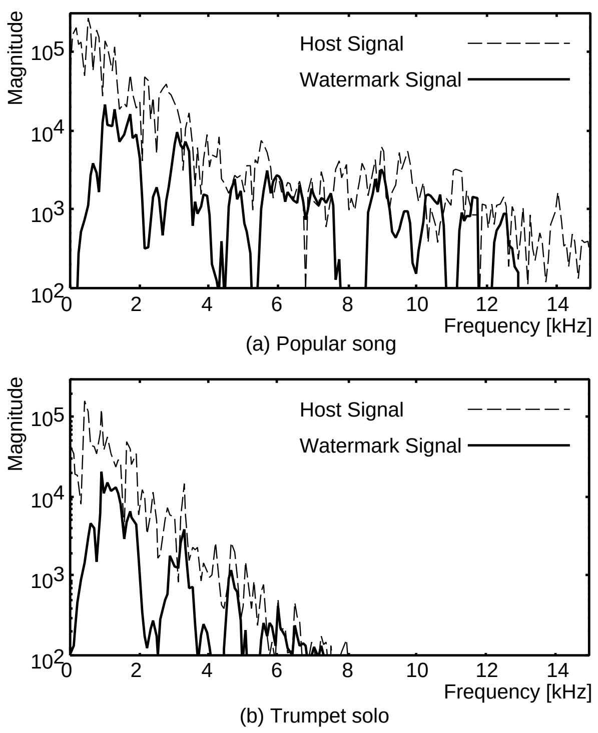

Figure 5.28: Examples of the watermark signal and the corresponding host signal for (a) a popular song and (b) a trumpet solo.

5.3.3 Implementation of sonic watermarking

We used the same algorithms described in Section 5.2.3 for the experiments. We implemented a watermark generator that can generate sonic watermarks in real-time and a detector that can detect 64-bit messages in 30-second pieces of music A Pentium IV 2.2 GHz Windows XP personal computer (PC) equipped with a Sound Blaster Audigy Platinum sound card by Creative Technology, Ltd. was used for the platform. The message is encoded in 448 bits by adding 8 Cyclic Redundancy Check (CRC) parity bits, using Turbo Coding, and repeating it twice. Each pattern block has 3 bits and a synchronization signal embedded, and the block has 24 columns and 8 rows of tiles. Each of the 24 frequency subbands is given an equal bandwidth of 6 frequency bins. The frequency of the highest bin used is 12.7 kHz. The length of a DFT frame is 512 samples to shorten the delay. Based on the psychoacoustic model, the root mean square power of the WS is 23.0 dB lower than that of the HS on average. Examples of watermark signals generated for a popular song and a trumpet solo are shown in Fig. 5.28.

At the time of detection, while 48 tiles out of the 192 tiles are dedicated for the local adjustment of the pattern block synchronization, the tiles assigned for the bits are also used for the global synchronization. For the global synchronization, It is assumed that 16 consecutive blocks have consistent synchronization positions. The false alarm error ratio is theoretically under 10^{-5} based on the threshold of the square means of the detected bit strengths. Another threshold on the estimated watermark signal-to-noise ratio (SNR) is set to keep the code word error ratio under 10^{-5}. The reasons to use both thresholds are described in Ref.[61].

Delay The delay of the WS was approximately 17.8 ms in total. The details are as follows. A total of 128 samples for both the playback buffer and the recording buffer were required

Table 5.3: The number and the durations of the test samples used for the robustness tests.

Category	# of samples	Duration
Popular Music	20	92 min
Orchestral Music	13	112 min
Instrumental Solos	76	120 min

for stable real-time watermark generation. The length of a DFT frame was 512 samples. The watermark calculation process took approximately 3.1% of the playback time. Since the length of a DFT frame was 512 samples, the elapsed time for the WS calculation corresponds approximately to the playback time for 16 samples. Hence, the total delay was $128 + 128 + 512 + 16 = 784$ samples, which was about 17.8 ms for 44.1 kHz sampling.

5.3.4 Results of robustness tests

We tested the robustness of the algorithm against transformations that are important for the lifecycle of sonic watermarking: sonic propagation, echo addition, noise addition, and MP3 compression. The results of the tests were collected for three categories: (a) popular music, (b) orchestral music, and (c) instrumental solos. The numbers of test samples and the duration for each category are listed in Table 5.3. The test samples of instrumental solos included 59 samples of performance of single instruments from SQAM[8]. All of the signals were monaural and sampled at a frequency of 44.1 kHz and with a bit resolution of 16 bits. Since it has been shown in Ref. [66] that real-time sonic watermarking using the proposed algorithm is feasible, we did not use real-time watermarking for the tests. We calculated the WS off-line, and added them to or played them simultaneously with the HS.

(a) Results

We measured the correct detection rates (CDRs) at which the correct 64-bit messages were detected. The error correction and detection algorithm successfully avoided detection of an incorrect message.

Robustness against MP3 compression Table 5.4 shows the results for sonic watermarking and MP3 compression. "Digital WM" means that the WS was digitally added to the HS with a delay of 20 ms. "Sonic WM" means that the sound of the WS was mixed with the host sound in the air and recorded by a microphone. We used the same experimental equipment as used for sd20 of the listening test. For the "Original watermark", the watermark was detected immediately after watermark embedding as described above. For "MP3", the watermarked signal was compressed in a MP3 file with the specified bit rate for a monaural channel and then decompressed before watermark detection. For popular music and orchestral music, correct watermarks were detected from over 95% of detection windows after sonic watermarking and MP3 compression. The reason the CDRs for instrumental solos were low is that the test samples included many sections that are almost silent or at a quite low volume, and the watermarks in those sections were easily destroyed by the background noise of the room and by the MP3 compression. We observed a 28 dB(A)[9] background noise in the soundproof room when nothing was played by the speakers.

[8]Sound Quality Assessment Material disc produced by the European Broadcasting Union for subjective tests.

[9]dB(A) is a unit for the A-weighted sound level [16].

Table 5.4: The correct detection rates (CDRs) at which the correct 64-bit messages were detected. Watermark embedding was performed by digital addition (Digital WM) or sonic watermarking (Sonic WM). Detection was done immediately after embedding or after MP3 compression and decompression.

Popular Music	Digital WM	Sonic WM
Original watermark	100%	96%
MP3 64 kbps	100%	96%
MP3 48 kbps	100%	95%
Orchestral Music	**Digital WM**	**Sonic WM**
Original watermark	100%	99%
MP3 64 kbps	100%	99%
MP3 48 kbps	100%	97%
Instrumental Solos	**Digital WM**	**Sonic WM**
Original watermark	99%	60%
MP3 64 kbps	97%	53%
MP3 48 kbps	66%	37%

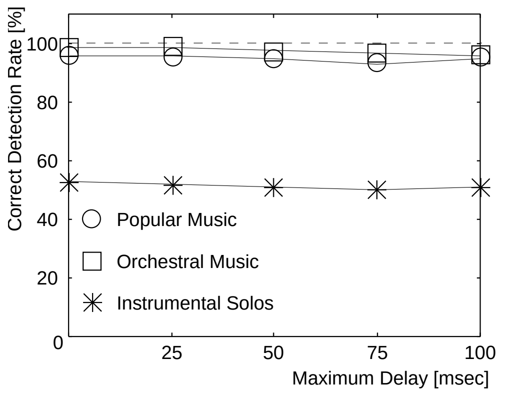

Figure 5.29: The correct detection rates (CDRs) after sonic watermarking and echo addition. The leftmost points are the rates immediately after sonic watermarking.

Robustness against echo addition Figure 5.29 shows the CDRs after sonic watermarking and echo addition. Echoing was done digitally on a computer with a feedback coefficient of 0.5. The horizontal axis of the figure is the value of the maximum delay used for echo addition. Although the CDRs for the instrumental solos were low because of sonic watermarking, it can be seen that echo addition interferes very little with watermark detection.

Robustness against noise addition Figure 5.30 shows the CDRs after sonic watermarking and noise addition. White Gaussian noises with an average noise-to-signal ratio shown in the horizontal axis of the figure were digitally added to the recordings. For popular music, the CDRs remained high up to -20 dB of noise addition. In contrast, the CDRs for orchestral music dropped after noise addition above -35 dB. This is because orchestral music has wider dynamic ranges than popular music does, and contains more low volume sections. Those quiet sections degrade more quickly than loud sections do when the additive noise has a comparable signal level. Although it has been shown in Ref. [66] that correct detection rate for quiet

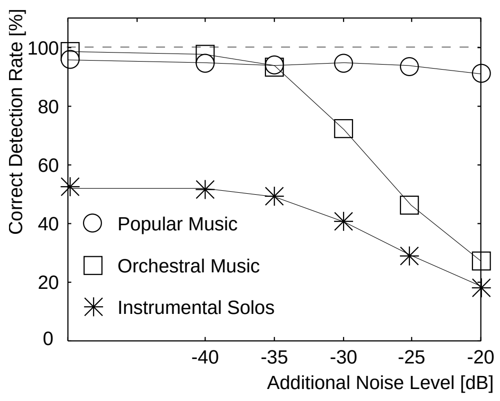

Figure 5.30: The CDRs after sonic watermarking and noise addition. The leftmost points are the rates immediately after sonic watermarking.

sections can be improved, at the sacrifice of acoustic quality, by utilizing the masking effect of the background noise, the robustness against noise when the masking effect is not used by the watermark generator is still an open problem.

5.3.5 Results of subjective listening tests

The evaluation of the subjective audio quality of the algorithm was done by a MUSHRA [31] listening test. The effects of two factors that can be considered to be particularly important for the use of sonic watermarking are also investigated. Those are (1) the delay of the WS relative to the HS, and (2) the angle between the sound sources of the WS and the HS (as measured from the listener's location).

The test samples were monaural excerpts from popular music, orchestral music, and instrumental solos as described in Table 5.5. The mean duration of the samples was 12.3 sec. All of the test signals were sampled at a frequency of 44.1 kHz and with a bit resolution of 16 bits. All of them were upsampled to 48 kHz before the test to adjust to the listening equipment. Although most of the 18 subjects were inexperienced listeners, there were training sessions in advance of the test in which they were exposed to the full range and nature of all of the test signals. To give anchors for comparison, the subjects were also required to assess the audio quality of hidden references (**hr**)[10], 7 kHz low-pass filtered samples (**al7**), and samples which had been compressed in MP3 files with a bit rate of 48 kbps (**am48**) or 64 kbps (**am64**) for a monaural channel using the Fraunhofer codec of MUSICMATCH Jukebox 7.20. The references (**r**), the hidden references, and the anchors were played by the speaker SP1 (Fig. 5.31). The other test signals (Table 5.6) were as described below:

sd10 Sonic watermark with a delay of 10 ms

While the HS completely identical to the reference was played from SP1, a WS that had been computed in advance based on the HS was simultaneously played from another speaker, SP2, with a delay of 10 ms. SP2 was offset from the direction of SP1 by 4.3°. The subjects listened to the mixed sound of the HS and the WS.

sd20 Sonic watermark with a delay of 20 ms

The same WS used for sd10 was played from SP2 with a delay of 20 ms, which is close to the delay of our implementation.

[10]Although the test signals of the hidden references were identical to the reference signals, the subjects were required to assess their quality without knowing which were which.

Table 5.5: The test samples for the listening tests.

Sample	Duration	Category	Description
is1	8 sec	Solo	Castanets
is2	10 sec	Solo	Glockenspiel
is3	12 sec	Solo	Guitar
is4	14 sec	Solo	Trumpet
io1	15 sec	Orchestra	Soloists and orchestra
io2	12 sec	Orchestra	Wind ensemble
ip1	16 sec	Popular	Eddie Rabbitt
ip2	13 sec	Popular	Michael Jackson
ip3	12 sec	Popular	Mai Kuraki

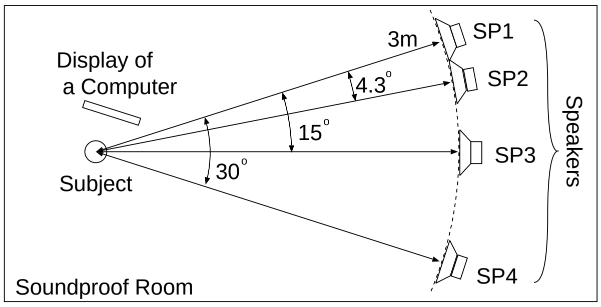

Figure 5.31: The listening environment for the MUSHRA subjective listening tests. Three speakers, SP2 to SP4, were at offsets from the direction of SP1 by 4.3°, 15°, and 30°, respectively.

sd40 Sonic watermark with a delay of 40 ms
The WS was played from SP2 with a delay of 40 ms.

sa15 Sonic watermark with an angle of 15°
The WS was played from another speaker, SP3, with a delay of 20 ms. SP3 was offset 15° from SP1.

sa30 Sonic watermark with an angle of 30°
The WS was played from another speaker, SP4, with a delay of 20 ms. SP4 was offset 30° from SP1.

(a) Results

The mean and 95% confidence interval of the subjective acoustic quality of the test signals are shown in Fig. 5.32. The quality of sonic watermarks with a delay equal to or less than 20 ms was assessed in the range of *excellent quality.* Although the WSs were not inaudible, the acoustic quality for most of the test samples can be considered to be good enough for the realistic use.

Effect of the delay The relationship of the quality and the delay is shown in Fig. 5.33. Most subjects could notice acoustic impairments in sd40 and reduced its score to *Good quality.* Especially in the case of Castanets (Fig. 5.34), the watermark sound with a large delay could be heard as additional small castanets. A similar effect also occurred for drumbeats and cymbals

Table 5.6: The test signals for the listening tests; SP1 to SP4 are the speakers illustrated in Fig. 5.31. Monaural signals simultaneously played from the speakers are listed in this table. The abbreviations are explained in Table 5.7.

Signal	SP1	SP2	SP3	SP4
r	REF	–	–	–
hr	REF	–	–	–
am64	$MP3_{64}$	–	–	–
am48	$MP3_{48}$	–	–	–
al7	LP7	–	–	–
sd10	REF	WD10	–	–
sd20	REF	WD20	–	–
sd40	REF	WD40	–	–
sa15	REF	–	WD20	–
sa30	REF	–	–	WD20

Table 5.7: Description of the abbreviations used in Table 5.6.

Abbrev.	Description
REF	Reference monaural signal
$MP3_{64}$	Compressed signal using MP3 64 kbps
$MP3_{48}$	Compressed signal using MP3 48 kbps
LP7	7 kHz low-pass filtered signal
WD10	Watermark signal with 10 ms delay
WD20	Watermark signal with 20 ms delay
WD40	Watermark signal with 40 ms delay

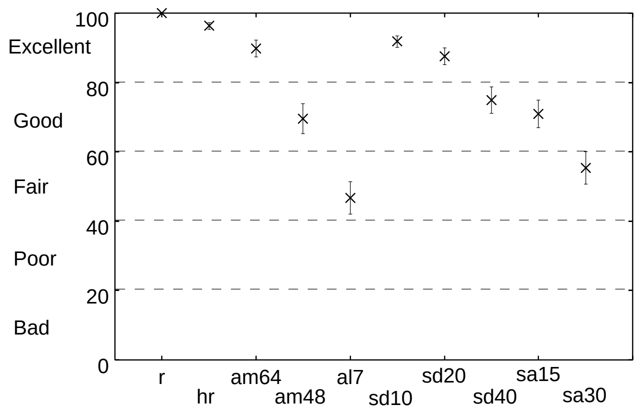

Figure 5.32: The mean and 95% confidence interval of the subjective acoustic quality of the test signals for all subjects. The test signals are described in Table 5.6.

in the popular music (Fig. 5.35). In those cases, the subjects perceived increased noisiness at the higher frequencies. For the test samples in which long notes were hold for some seconds (Fig. 5.36), the effect of the delay was low. In general, the quality difference between sd10 and sd20 was assessed to be small, and subjects sometimes gave sd20 better evaluations than sd10.

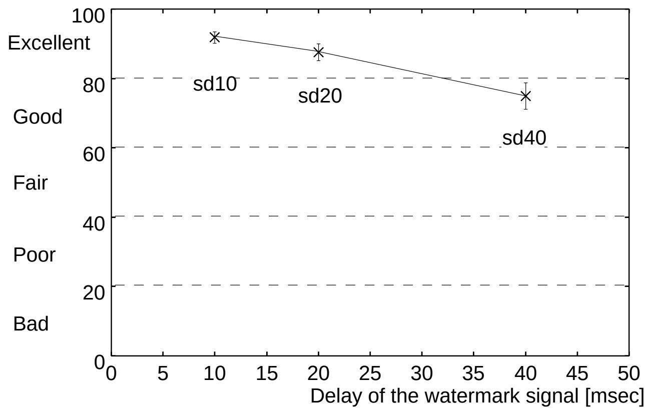

Figure 5.33: The relationship between the delay of the WS and the subjective acoustic quality.

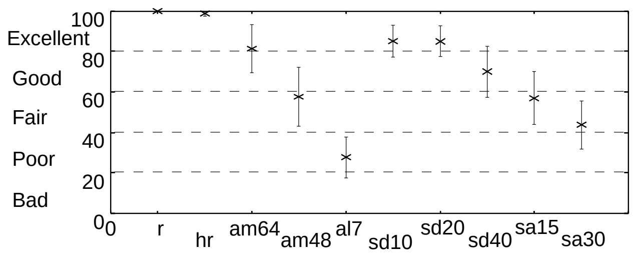

Figure 5.34: The subjective acoustic quality of the instrumental solo test sample is1, *Castanets.*

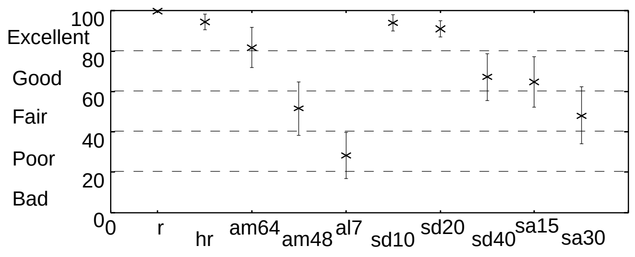

Figure 5.35: The subjective acoustic quality of the popular music test sample ip3, *Mai Kuraki.*

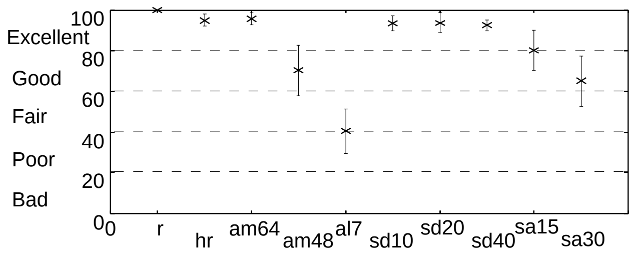

Figure 5.36: The subjective acoustic quality of the orchestral music test sample io2, *Wind ensemble.*

Effect of the sound source direction The relationship of the quality and the sound source direction is shown in Fig. 5.37. The effect was so large that sa30 was assessed in the range of *Fair*. When the WS was played from SP4, subjects noticed the difference by perceiving a weak stereo effect. However, in the case of sd20, even though the WS was played from SP2 in addition to the HS from SP1, subjects perceived the mixed sound as a monaural sound. The

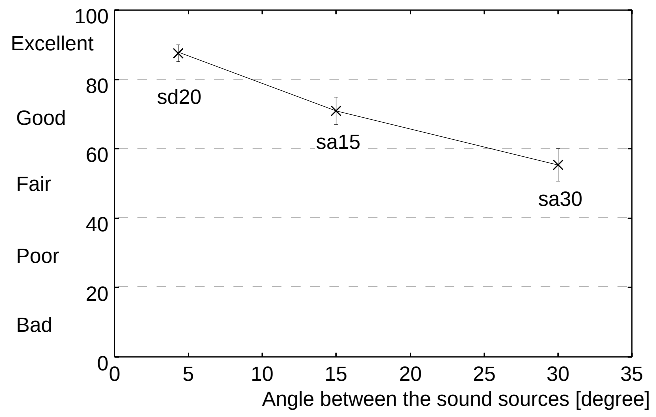

Figure 5.37: The relationship between the offset angle of the sound sources and the subjective acoustic quality.

effect was particularly prominent for the test samples for which the effect of the delay was distinguishable. Although the situation would be more complicated with multiple sources of the host sound for the realistic use of sonic watermarking, the experimental results suggest the sound source of the WS should be placed as close to the source of the host sound as possible.

In this section, we classified the possible problems that may limit the use of sonic watermarking. The subjective acoustic quality of the algorithm was assessed in the range of *excellent quality* by the MUSHRA listening test. We assessed the effect of the delay of the watermark signal on the quality, and found that 20 ms was enough short to sustain excellent quality. The effect of the direction of the sound sources of the watermark signal and the host signal was so large that special attention should be paid to the placement of the sound sources when using sonic watermarking. The experimental results of robustness were dependent on the type of the music samples. For popular music, the watermark was quite robust, so that correct messages were detected from over 90% of the detection windows even when noise addition, echo addition, or MP3 compression was performed after sonic watermarking. However, in the case of instrument solos, since the watermarks for low volume sections were easily degraded by the background noise, the correct detection rate after sonic watermarking was only 60%.

5.4 Concluding Remarks

In this chapter, we showed that the audio watermarking algorithm described in the previous chapter can be applied to broad range of applications. First, we showed the algorithm was applicable to compressed audio in the MPEG2 Advanced Audio Coding (AAC) format. This was possible because the algorithm relies only on the magnitudes of the audio. Experimental results showed that the watermark embedded in an uncompressed audio file was successfully detected in the AAC-compressed domain after the file is compressed and that the watermark embedded in an AAC file was able to be detected after the file was decompressed, trimmed, and compressed again.

Although we have not mentioned in the dissertation, the basic algorithm is also applicable to broadcast music monitoring. Precise determination of the start time and the finish time of each playback is required for broadcast music monitoring. We also showed, in [47], that this is also possible with the basic algorithm by sliding the detection time range little by little and detecting watermark from each of the time ranges.

As a natural extension of broadcast music monitoring, there are needs for real-time watermarking of live broadcast. In response to this kind of needs, we introduced various composition methods for real-time audio watermark embedding and showed how they can extend the range of applications of audio watermarks. In a composition method named *Analog Watermarking*, a trusted conventional analog mixer is used to mix the host signal (HS) and the watermark signal (WS) after the WS is generated by a computer and converted to an analog signal.

We also introduced the idea of sonic watermarking that mixes the sound of the watermark signal and the host sound in the air to detect bootleg recordings. The possible problems that may limit the use of sonic watermarking were classified. We discussed an audio watermarking algorithm suitable for sonic watermarking. The subjective acoustic quality of the algorithm was assessed in the range of *excellent quality* by the MUSHRA listening test. The experimental results of robustness were dependent on the type of the music samples. For popular music, the watermark was quite robust, so that correct messages were detected from over 90% of the detection windows even when noise addition, echo addition, or MPEG1 Audio Layer 3 (MP3) compression was performed after sonic watermarking. However, in the case of instrument solos, since the watermarks for low volume sections were easily degraded by the background noise, the correct detection rate after sonic watermarking was only 60%.

Furthermore, by using the robustness of the basic algorithm against sound propagation, we also showed in [50, 48] that it is possible to determine the recording location of bootleg recordings. This is achieved by embedding a different watermark signal into each of the multiple channels of the host signal.

However, there are still large problems to solve with these methods. When the basic algorithm is applied to compressed audio or sound in the air, the robustness and acoustic quality are worse compared to the previous embedding method. Although the acoustic quality of sonic watermarking was assessed in the range of *excellent quality*, we can see from the experimental results that human ears was able to distinguish the sonic-watermarked sound from the original sound. Further improvement of the algorithm is required to achieve better robustness and higher acoustic quality especially when it is applied to compressed audio or sound in the air.

Chapter 6

Conclusion

Although many procedures and systems are increasingly automated, the slow automation of the sound-based human-computer interface is a bottleneck for the automation of entire systems and diminishes the benefits of those systems. Therefore, we are aiming to develop audio processing technologies that enable automatic exchanges of audio data between the computers and the real world, while preserving the naturalness of the sound and the naturalness of the human-computer interactions that use sound. Among various audio processing technologies, we focused on speech synthesis and audio watermarking, since both are currently focuses of attention for research and are in high demand for applications. We introduced some application systems using these technologies in Chapter 2 to clarify the requirements for the audio processing technologies. By describing the roles of the technologies in the systems, the importance of the acoustic quality of the technologies was reaffirmed. No matter what convenience such technologies offer, people will not willingly use these technologies if the acoustic quality is not satisfactory.

Because the training of accurate stochastic models is important for the acoustic quality of text-to-speech (TTS), we focused our TTS work on accurate and automatic training of the stochastic models. For this purpose, we introduced, in Chapter 3, a totally trainable TTS system, every component of which can be automatically built from human speech alone. In this system, the necessary linguistic and acoustic information was incrementally collected by combining acoustic processing and linguistic processing. As a sub-module of the system, the automatic prosody labeling algorithm achieved an F measure of 0.862 for prosodic phrase boundary detection by using the linguistic information for part-of-speech (POS) with acoustic features, which was better than the results reported in previous research. For the accent determination problem of Japanese, we achieved 92.7% mora accuracy by using an n-gram linguistic model, including the spelling of the words. The overall quality of the synthetic voices was evaluated by subjective listening tests. It was seen from the results that the quality of the synthetic voices for in-domain text was over 4.0 out of a possible score of 5 in the subjective tests and approached the quality of recorded human voices. However, the acoustic quality for out-of-domain text was assessed to be far worse than the acoustic quality for in-domain text.

For improving the acoustic quality of audio watermarking, we need to improve the trade-off balance points among the acoustic quality, robustness, and data payload of the watermarking algorithms. Therefore, we presented a robust audio watermarking method using a two-dimensional pseudo-random array in Chapter 4. We tested the robustness of the method against various kinds of audio processing. The correct detection rates (CDRs) for 96 kpbs MPEG1 Audio Layer 3 (MP3) compression, 100-msec echo addition, and -30dB noise addition were 100%, 97%, and 87%, respectively. In addition, by detecting with multiple stretched patterns, the CDRs for -10% pitch shifting, +10 pitch shifting, -10% random stretching, and

+10% random stretching were 100%, 100%, 83%, and 87%, respectively. Although we do not have any results of formal subjective listening tests to present in this dissertation, the robustness experiments were performed with the watermark signal level 35dB lower than the level of the host signal, which is a level that is virtually indistinguishable to human ears. The acoustic quality of the method was verified also through a number of informal subjective listening tests. In these tests, professional sound engineers and audio equipment experts listened carefully to watermarked audio samples. The theoretical communication capacities of the algorithm under conditions with additive noise were formulated. The maximum communication capacity for pop music and orchestral music per 30-second excerpts was approximately 400 bits and 100 bits, respectively. The reason why the communication capacity differs for each genre of audio content is that the different distributions of the magnitudes caused different degrees of robustness against additive noise.

Chapter 5 discussed the audio watermarking algorithm can be applied to a broad range of applications. The first application area was compressed audio. We aimed to allow watermark detection in both the compressed domain and the uncompressed domain regardless of the original domain where the watermark embedding was done. When the watermark was embedded in the uncompressed domain, CDRs over 80% were seen for every test case. The most difficult challenge was watermark detection in the compressed domain when the content was first watermarked in the compressed domain, then decompressed to the uncompressed domain, and finally compressed again. The CDRs for detection in the re-compressed audio files were 60-80%. The second application area was for live performances. We introduced the various composition methods for real-time audio watermark embedding and showed how they can extend the range of applications of audio watermarking. We also introduced the idea of sonic watermarking that mixes the sound of the watermark signal and the host sound in the air to detect bootleg recordings. The subjective acoustic quality of the algorithm was assessed in the range of *excellent quality* by the MUSHRA listening test. For popular music, the watermark was quite robust, so that correct messages were detected from over 90% of the detection windows even when MP3 compression was performed after sonic watermarking. However, in the case of instrumental solos, since the watermarks for quiet passages were easily degraded by the background noise, the correct detection rate after sonic watermarking was only 60%.

Reviewing these results, we can see that the research results described in this dissertation were achieved based upon a few initial important core ideas and other research items flowed naturally from the core ideas. It is worth identifying the core ideas to consolidate the insights into the problems. Our core idea for text-to-speech synthesis was stochastic text processing. Compared to the automatic speech recognition research area, the use of stochastic language models was delayed in the conventional TTS research. It had been common to use rule-based text processing modules that could not handle the speaker dependencies of pronunciations and prosodic labels. The introduction of the stochastic text processing made all of the major modules of the TTS system trainable, and enabled us to design the $\mathrm{T}^4\mathrm{S}$ framework. The text processing module was also effective for accuracy improvements of the sub-modules of the framework, the POS tagging and the automatic prosody labeling components, and it resulted in a better final acoustic quality. In addition, since it became possible to make the text processing module speaker-dependent, the reproduction of pronunciations and prosodic labels of a specific speaker was possible. The most important core idea for audio watermarking was to modify magnitude differences based on a two-dimensional pseudo-random array in the frequency domain, while the previously standard audio watermarking approach was to embed a phrase-dependent watermark in the time domain exploiting human audio insensitivity to phase changes. Our new approach provided a high degree of control of the acoustic quality in the frequency domain by direct use of a psychoacoustic model. The watermark embedding in magnitude differences makes watermarks that are less subject to the transfer characteristics of the systems in which the watermarked audio is transferred. Since this allowed us to cover various systems and formats as the objectives of audio watermarking, we were able to naturally think of new applications and the associated improvements. If the initial core ideas had been in the wrong direction or had not been essentially significant, we would not have been able to extend these research projects. In addition, we should not forget that the demands of the markets were the driving forces both for the core ideas and for the extended research items.

While we were able to achieve the target levels of acoustic quality, accuracy, and robustness for some input and use cases, there remain cases of text or music where we could not achieve the target levels. For example, the results of the subjective listening tests of synthetic voices showed that, while the score for in-domain text was over 4.0, the score for out-of-domain text was worse than the acoustic quality for in-domain text. The degradation of the sound by sonic watermarking was also noticeable in some cases. Further improvement is certainly required for these applications. To improve the acoustic quality of the synthetic voices for out-of-domain text, we need to further improve the accuracy of each of the sub-modules of the T^4S framework. The use of more global features may be effective for this objective. Distinguishing between primary accents and secondary accents will refine the accent models. Since methods for unsupervised training have been proposed in the research area of automatic speech recognition (ASR), it is desirable that these methods be applied to the T^4S framework. Regarding audio watermarking, a problem when applying the algorithms to sonic watermarking is that the algorithms cannot effectively modify the magnitudes of the audio content when the watermark signal is delayed relative to the host signal. An improved technique to address this problem is necessary. We hope research in this direction will soon enable more natural human-computer interactions using sound.

Bibliography

[1] M. Abe, Y. Sagisaka, T. Umeda, and H. Kuwabara. "Speech Database User's Manual," Technical report, ATR TR-I-0166, 1990.

[2] J. Adell, P. D. Agüero, and A. Bonafonte. "Database Pruning for Unsupervised Building of Text-to-Speech Voices," In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages I–889–I–892, May 2006.

[3] S. Ananthakrishnan and S. Narayanan. "An Automatic Prosody Recognizer Using a Coupled Multi-stream Acoustic Model and a Syntactic-prosodic Language Model," In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages I–269–I–272, March 2005.

[4] M. Arnold and S. Kanka. "MP3 Robust Audio Watermarking," In *DFG $V^{III}D^{II}$ Watermarking Workshop*, 1999.

[5] M. Barni, F. Bartolini, A. D. Rosa, and A. Piva. "Optimum Decoding and Detection of A Multiplicative Amplitude Encoded Watermark," In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents IV*, pages 409–420, January 2001.

[6] M. Barni, F. Bartolini, A. De Rosa, and A. Piva. "Capacity of the Watermark-channel: How Many Bits Can Be Hidden Within a Digital Image?," In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents*, pages 437–448, January 1999.

[7] P. Bassia and J. Pitas. "Robust Audio Watermarking in the Time Domain," In *Proc. European Signal Processing Conference (EUSIPCO)*, volume 1, pages 25–28, September 1998.

[8] W. Bender, D. Gruhl, and N. Morimoto. "Techniques for Data Hiding," Technical report, MIT Media Lab, 1994.

[9] W. Bender, D. Gruhl, N. Morimoto, and A. Lu. "Techniques for Data Hiding," *IBM Systems Journal*, 35(3-4):313–336, 1996.

[10] L. Boney, A. H. Tewfik, and K. N. Hamdy. "Digital Watermarks for Audio Signals," In *Proc. IEEE Int. Conf. on Multimedia Computing and Systems*, pages 473–480, June 1996.

[11] N. Campbell. "Autolabelling Japanese ToBI," In *Proc. Int. Conf. on Spoken Language Processing (ICSLP)*, pages 2399–2402, October 1996.

[12] K. Chen, M. Hasegawa-Johnson, and A. Cohen. "An Automatic Prosody Labeling System Using ANN-based Syntactic-prosodic Model and GMM-based Acoustic-prosodic Model," In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages I–509–I–512, May 2004.

[13] S. Cheng, H. Yu, and Z. Xiong. "Enhanced Spread Spectrum Watermarking of MPEG-2 AAC Audio," In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages IV–3728–IV–3731, May 2002.

[14] A. Conkie, G. Riccardi, and R. C. Rose. "Prosody Recognition from Speech Utterances Using Acoustic and Linguistic Based Models of Prosodic Events," In *Proc. Eurospeech*, pages 523–526, September 1999.

[15] I. J. Cox, M. L. Miller, and J. A. Bloom. *Digital Watermarking.* Morgan Kaufmann, 2001.

[16] M. J. Crocker. "Rating Measures, Descriptors, Criteria, and Procedures for Determining Human Response to Noise," In M. J. Crocker, editor, *Encyclopedia of Acoustics*, volume 2, chapter 80, pages 943–965. John Wiley & Sons, Inc., 1997.

[17] D. Delannay, J. F. Delaigle, B. Macq, and M. Barlaud. "Compensation of Geometrical Deformations for Watermark Extraction in the Digital Cinema Application," In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents III*, volume 4314, pages 149–157, January 2001.

[18] J. Dittmann, M. Steinebach, and R. Steinmetz. "Digital Watermarking for MPEG Audio Layer 2," In *Proc. Workshop of Multimedia and Security at ACM Multimedia*, pages 117–122, October 1999.

[19] R. E. Donovan and P. C. Woodland. "Improvements in an HMM-Based Synthesizer," In *Proc. Eurospeech*, pages 573–576, September 1995.

[20] E. Eide, A. Aaron, R. Bakis, P. Cohen, R. Donovan, W. Hamza, T. Mathes, M. Picheny, M. Polkosky, M. Smith, and M. Viswanathan. "Recent Improvements to the IBM Trainable Speech Synthesis System," In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages I–708–I–711, April 2003.

[21] K. Emoto, H. Zen, K. Tokuda, and T. Kitamura. "Accent Type Recognition for Automatic Prosodic Labeling," In *Proc. Fall Meeting of Acoustic Society of Japan*, pages 225–226, September 2003 (in Japanese).

[22] R. Garcia. "Digital Watermarking of Audio Signals Using a Psychoacoustic Auditory Model and Spread Spectrum Theory," In *Proc. 107th Convention, Audio Engineering Society, Preprint 5073*, 1999.

[23] D. Gruhl, A. Lu, and W. Bender. "Echo Hiding," In *Information Hiding*, pages 293–315, 1996.

[24] J. Haitsma, M. van der Veen, F. Bruekers, and T. Kalker. "Audio Watermarking for Monitoring and Copy Protection," In *Proc. ACM Multimedia*, pages 119–122, November 2000.

[25] K. Hirose and K. Iwano. "Accent Type Recognition and Syntactic Boundary Detection of Japanese Using Statistical Modeling of Moraic Transitions of Fundamental Frequency Contours," In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages I–25–I–28, May 1998.

[26] K. Hirose and K. Iwano. "Detection of Prosodic Word Boundaries by Statistical Modeling of Mora Transitions of Fundamental Frequency Contours and Its Use for Continuous Speech Recognition," In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages III–1763–III–1766, June 2000.

[27] H. Hon, A. Acero, X. Huang, J. Liu, and M. Plumpe. "Automatic Generation of Synthesis Units for Trainable Text-to-Speech Systems," In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages I–293–I–296, May 1998.

[28] X. Huang, A. Acero, H. Hon, Y. Ju, J. Liu, S. Meredith, and M. Plumpe. "Recent Improvements on Microsoft' s Trainable Text-to-Speech System - Whistler," In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages II–959–II–962, April 1997.

[29] "ISO/IEC 11172-3:1993: Information Technology – Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/S – Part 3: Audio," 1993.

[30] "ISO/IEC 13818-7:1997: Information Technology – Generic Coding of Moving Pictures and Associated Audio Information – Part 7: Advanced Audio Coding (AAC)," 1997.

[31] "ITU-R Recommendation BS.1534-1: Method for the Subjective Assessment of Intermediate Quality Level of Coding Systems," 2003.

[32] K. Iwano and K. Hirose. "Representing Prosodic Words Using Statistical Models of Moraic Transition of Fundamental Frequency Contours," In *Proc. Int. Conf. on Spoken Language Processing (ICSLP)*, pages 599–602, December 1998.

[33] Japan Society for the Rights of Authors, Composers and Publishers. "Announcement of Evaluation Test Results for STEP 2000, International Evaluation Project for Digital Watermark Technology for Music," http://www.jasrac.or.jp/watermark/ehoukoku.htm, October 2000.

[34] Japan Society for the Rights of Authors, Composers and Publishers. "Final Selection of Technology toward the Global Spread of Digital Audio Watermarks," http://www.jasrac.or.jp/step2001/presse.htm, October 2001.

[35] D. Kirovski and H. Malvar. "Robust Covert Communication over a Public Audio Channel Using Spread Spectrum," In *Proc. 4th Int. Workshop on Information Hiding*, volume LNCS 2137, pages 354–368, April 2001.

[36] D. Kirovski and H. Malvar. "Robust Spread-spectrum Audio Watermarking," In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, pages III–1345–III–1348, May 2001.

[37] D. Kirovski and H. Malvar. "Spread-spectrum Audio Watermarking: Requirements, Applications, and Limitations," In *Proc. IEEE Forth Workshop on Multimedia Signal Processing*, pages 219–224, October 2001.

[38] J. Kominek and A. W Black. "Impact of Durational Outlier Removal from Unit Selection Catalogs," In *Proc. 5th ISCA Speech Synthesis Workshop*, pages 155–160, June 2004.

[39] D. K. Koukopoulos and Y. C. Stamatiou. "A Compressed-domain Watermarking Algorithm for MPEG Audio Layer 3," In *Proc. ACM Multimedia*, October 2001.

[40] C.-C. Kuo, C.-S. Kuo, J.-H. Chen, and S.-C. Chang. "Automatic Speech Segmentation and Verification for Concatenative Synthesis," In *Proc. Eurospeech*, pages 305–308, September 2003.

[41] M. Kutter. "Watermarking Resisting to Translation, Rotation, and Scaling," In *Proc. SPIE Int. Conf. on Multimedia Systems and Applications*, volume 3528, pages 423–431, January 1998.

[42] M. Lai, Y. Chen, M. Chu, Y. Zhao, and F. Hu. "A Hierarchical Approach to Automatic Stress Detection in English Sentences," In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages I–753–I–756, May 2006.

[43] C.-Y. Lin, M. Wu, J. A. Bloom, I. J. Cox, M. L. Miller, and Y.-M. Lui. "Rotation, Scale, and Translation Resilient Public Watermarking for Images," In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents II*, volume 3971, pages 90–98, January 2000.

[44] T. Nagano, S. Mori, and M. Nishimura. "A Stochastic Approach to Phoneme and Accent Estimation," In *Proc. Interspeech*, pages 3293–3296, September 2005.

[45] T. Nagano, R. Tachibana, S. Mori, and M. Nishimura. "An Improvement of a Stochastic-based Front-end for Text-to-Speech System," In *Proc. Spring Meeting of Acoustic Society of Japan*, pages 327–328, March 2007 (in Japanese).

[46] M. Nakai, S. Harald, Y. Sagisaka, and H. Shimodaira. "Automatic Prosodic Segmentation by F0 Clustering Using Superpositional Modeling," In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages I–624–I–627, May 1995.

[47] T. Nakamura, R. Tachibana, and S. Kobayashi. "Automatic Music Monitoring and Boundary Detection for Broadcast Using Audio Watermarking," In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents IV*, volume 4675, pages 170–180, January 2002.

[48] Y. Nakashima, R. Tachibana, M. Nishimura, and N. Babaguchi. "Estimation of Recording Location Using Audio Watermarking," In *Proc. the 8th Workshop on Multimedia and Security*, pages 108–113, September 2006.

[49] Y. Nakashima, R. Tachibana, M. Nishimura, and N. Babaguchi. "Recording Device Localization Using Multiple Audio Watermarks," In *Proc. Fall Meeting of Acoustic Society of Japan*, pages 453–454, September 2006 (in Japanese).

[50] Y. Nakashima, R. Tachibana, M. Nishimura, and N. Babaguchi. "Determining Recording Location Based on Synchronization Positions of Audio Watermarking," In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages II–253–II–256, April 2007.

[51] C. Neubauer and J. Herre. "Digital Watermarking and Its Influence on Audio Quality," In *Proc. 105th Convention of Audio Engineering Society (AES)*, September 1998.

[52] C. Neubauer and J. Herre. "Audio Watermarking of MPEG2 AAC Bitstreams," In *Proc. 108th Convention of Audio Engineering Society (AES)*, February 2000.

[53] C. Neubauer, R. Kulessa, and J. Herre. "A Compatible Family of Bitstream Watermarking Schemes for MPEG-Audio," In *Proc. 110th Convention of Audio Engineering Society (AES)*, May 2001.

[54] C. Neubauer, M. Steinebach, F. Siebenhaar, and J. Pickel. “Robustness Evaluation of Transactional Audio Watermarking Systems,” In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents V*, volume 5020, pages 12–20, January 2003.

[55] M. Nishimura, R. Tachibana, T. Nagano, and G. Kurata. “A Study on Totally Trainable TTS System,” In *Proc. Fall Meeting of Acoustic Society of Japan*, pages 237–238, September 2006 (in Japanese).

[56] W. Wesley Peterson and E. J. Weldon Jr. *Error-correcting Codes.* MIT Press, 1988.

[57] L. Qiao and K. Nahrstedty. “Non-invertible Watermarking Methods for MPEG Encoded Audio,” In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents*, volume 3675, pages 194–202, January 1999.

[58] J. J. K. Ó Ruanaidh and T. Pun. “Rotation, Scale and Translation Invariant Spread Spectrum Digital Image Watermarking,” *Signal Processing*, 66:303–317, May 1998.

[59] Y. Sagisaka and H. Sato. “Accentuation Rules for Japanese Word Concatenation,” *Trans. IEICE Japan*, J66-D(7):849–856, 1983 (in Japanese).

[60] “Secure Digital Music Initiative (SDMI),” http://www.sdmi.org/.

[61] S. Shimizu. “Performance Analysis of Information Hiding,” In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents IV*, volume 4675, pages 421–432, January 2002.

[62] M. Steinebach and J. Dittmann. “Capacity-optimized MP2 Audio Watermarking,” In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents V*, volume 5020, pages 44–54, January 2003.

[63] M. D. Swanson, B. Zhu, A. H. Tewfik, and L. Boney. “Robust Audio Watermarking Using Perceptual Masking,” *Signal Processing*, 66:337–355, 1998.

[64] R. Tachibana. “Improving Audio Watermark Robustness Using Stretched Patterns Against Geometric Distortion,” In Y.-C. Chen and L.-W. Chang and C.-T. Hsu, editor, *Advances in Multimedia Information Processing*, Lecture Notes in Computer Science (LNCS) 2532, pages 647–654. Springer, December 2002. Proc. the 3rd IEEE Pacific-Rim Conf. on Multimedia (PCM2002).

[65] R. Tachibana. “Audio Watermarking for Realtime Embedding,” In *Proc. Forum on Information Technology (FIT)*, pages 297–298, September 2002 (in Japanese).

[66] R. Tachibana. “Audio Watermarking for Live Performance,” In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents V*, volume 5020, pages 32–43, January 2003.

[67] R. Tachibana. “Capacity Analysis of Audio Watermarking,” *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences (Japanese Edition)*, J86-A(11):1197–1206, November 2003 (in Japanese).

[68] R. Tachibana. “Sonic Watermarking,” *EURASIP Journal on Applied Signal Processing*, 2004(13):1955–1964, October 2004.

[69] R. Tachibana. “Two-dimensional Audio Watermark for MPEG AAC Audio,” In *Proc. SPIE Int. Conf. on Security, Steganography and Watermarking of Multimedia Contents VI*, volume 5306, pages 139–150, January 2004.

[70] R. Tachibana, T. Nagano, G. Kurata, M. Nishimura, and N. Babaguchi. “Automatic Prosody Labeling Using Multiple Models for Japanese,” *IEICE Trans. Information and Systems* (in press).

[71] R. Tachibana, T. Nagano, G. Kurata, M. Nishimura, and N. Babaguchi. “Automatic Accent Labeling for a Text-to-Speech System,” In *IPSJ SIG Technical Reports*, volume 2007-SLP-5, pages 97–102, February 2007.

[72] R. Tachibana, T. Nagano, G. Kurata, M. Nishimura, and N. Babaguchi. “Automatic Accent Labeling Using Multiple Models,” In *Proc. Spring Meeting of Acoustic Society of Japan*, pages 227–228, March 2007 (in Japanese).

[73] R. Tachibana, T. Nagano, G. Kurata, M. Nishimura, and N. Babaguchi. “Preliminary Experiments toward Automatic Generation of New TTS Voices from Recorded Speech Alone,” In *Proc. Interspeech*, August 2007 (in press).

[74] R. Tachibana, S. Shimizu, S. Kobayashi, and T. Nakamura. “An Audio Watermarking Method Using a Two-dimensional Pseudo-random Array,” *Signal Processing*, 82(10):1455–1469, October 2002.

[75] R. Tachibana, S. Shimizu, T. Nakamura, and S. Kobayashi. “An Audio Watermarking Method Robust against Time- and Frequency-fluctuation,” In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents III*, volume 4314, pages 104–115, January 2001.

[76] J. F. Tilki and A. A. Beex. “Encoding a Hidden Digital Signature onto an Audio Signal using Psychoacoustic Masking,” In *Proc. 7th Int. Conf. on Signal Processing Applications & Technology*, pages 476–480, October 1996.

[77] K. Tokuda, H. Zen, and A. W. Black. “An HMM-based Speech Synthesis System Applied to English,” In *Proc. IEEE Speech Synthesis Workshop*, September 2002.

[78] K. F. Tsang and O. C. Au. “Robust and High Quality Video Watermarking with the Use of Temporal Redundancy,” In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents III*, volume 4314, pages 55–63, January 2001.

[79] M. van der Veen, F. Bruekers, J. Haitsma, T. Kalker, A. Lemma, and W. Oomen. “Robust, Multifunctional and High-quality Audio Watermarking Technology,” In *Proc. 110th convention of Audio Engineering Society, Preprint 5345*, 2001.

[80] A. van Leest, J. Haitsma, and T. Kalker. “On Digital Cinema and Watermarking,” In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents V*, volume 5020, pages 526–535, January 2003.

[81] C. W. Wightman and M. Ostendorf. “Automatic Labeling of Prosodic Patterns,” *IEEE Transactions on Speech and Audio Processing*, 2(4):469–481, October 1994.

[82] C.-P. Wu, P.-C. Su, and C.-C. J. Kuo. “Robust and Efficient Digital Audio Watermarking Using Audio Content Analysis,” In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents II*, volume 3971, pages 382–392, January 2000.

[83] H. Zen, T. Toda, and K. Tokuda. "The Nitech-NAIST HMM-Based Speech Synthesis System for the Blizzard Challenge 2006," In *Proc. Blizzard Challenge 2006 workshop*, September 2006.

[84] E. Zwicker and H. Fastl. *Psychoacoustics.* Springer, 1999.

Publications

A Journal Papers

1. R. Tachibana, T. Nagano, G. Kurata, M. Nishimura, N. Babaguchi. "Automatic Prosody Labeling Using Multiple Models for Japanese," *IEICE Trans. Information and Systems* (in press).
2. R. Tachibana. "Sonic Watermarking," *EURASIP Journal on Applied Signal Processing*, 2004(13):1955-1964, October 2004.
3. R. Tachibana. "Capacity Analysis of Audio Watermarking," *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences (Japanese Edition)*, J86-A(11):1197-1206, November 2003 (in Japanese).
4. R. Tachibana, S. Shimizu, S. Kobayashi, and T. Nakamura. "An Audio Watermarking Method Using a Two-dimensional Pseudo-random Array," *Signal Processing*, 82(10):1455-1469, October 2002.

B International Conference Papers

1. R. Tachibana, T. Nagano, G. Kurata, M. Nishimura, N. Babaguchi. "Preliminary Experiments toward Automatic Generation of New TTS Voices from Recorded Speech Alone," In *Proc. Interspeech*, August 2007 (in press).
2. Y. Nakashima, R. Tachibana, M. Nishimura, and N. Babaguchi. "Determining Recording Location Based on Synchronization Positions of Audio Watermarking,", In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages II-253-II-256, April 2007.
3. Y. Nakashima, R. Tachibana, M. Nishimura, and N. Babaguchi. "Estimation of Recording Location Using Audio Watermarking," In *Proc. the 8th Workshop on Multimedia and Security*, pages 108-113, September 2006.
4. R. Tachibana. "Two-dimensional Audio Watermark for MPEG AAC Audio," In *Proc. SPIE Int. Conf. on Security, Steganography and Watermarking of Multimedia Contents VI*, volume 5306, pages 139-150, January 2004.
5. R. Tachibana. "Audio Watermarking for Live Performance," In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents V*, volume 5020, pages 32-43, January 2003.
6. R. Tachibana. "Improving Audio Watermarking Robustness Using Stretched Patterns against Geometric Distortion," In Y.-C. Chen and L.-W. Chang and C.-T. Hsu, editor, *Advances in Multimedia Information Processing*, Lecture Notes in Computer Science (LNCS) 2532, pages 647-654. Springer, December 2002. Proc. the 3rd IEEE Pacific-Rim Conference on Multimedia (PCM2002).

7. T. Nakamura, R. Tachibana, and S. Kobayashi. "Automatic Music Monitoring and Boundary Detection for Broadcast Using Audio Watermarking," In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents IV*, volume 4675, pages 170-180, January 2002.
8. R. Tachibana, S. Shimizu, S. Kobayashi, and T. Nakamura. "An Audio Watermarking Method Robust against Time- and Frequency-fluctuation", In *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents III*, volume 4314, pages 104-115, January 2001.

C Technical Papers

1. R. Tachibana, T. Nagano, G. Kurata, M. Nishimura, and N. Babaguchi. "Automatic Accent Labeling Using Multiple Models," In *Proc. Spring Meeting of Acoustic Society of Japan*, pages 227-228, March 2007 (in Japanese).
2. T. Nagano, R. Tachibana, S. Mori, and M. Nishimura. "An Improvement of a Stochastic-based Front-end for Text-to-Speech System,", In *Proc. Spring Meeting of Acoustic Society of Japan*, pages 327-328, March 2007 (in Japanese).
3. R. Tachibana, T. Nagano, G. Kurata, M. Nishimura, and N. Babaguchi. "Automatic Accent Labeling for a Text-to-Speech System," In *IPSJ SIG Technical Reports*, volume 2007-SLP-5, pages 97-102, Feburuary 2007.
4. Y. Nakashima, R. Tachibana, M. Nishimura, and N. Babaguchi. "Recording Device Localization Using Multiple Audio Watermarks," In *Proc. Fall Meeting of Acoustic Society of Japan*, pages 453-454, September 2006 (in Japanese).
5. M. Nishimura, R. Tachibana, T. Nagano, and G. Kurata. "A Study on Totally Trainable TTS System," In *Proc. Fall Meeting of Acoustic Society of Japan*, pages 237-238, September 2006 (in Japanese).
6. R. Tachibana. "Audio Watermarking for Realtime Embedding," In *Proc. Forum on Information Technology (FIT)*, pages 297-298, September 2002 (in Japanese).