Wavelet-based Audio Watermarking Techniques: Robustness and Fast Synchronization

Hong Oh Kim* Bae Keun Lee† Nam-Yong Lee‡

Abstract

This paper describes a novel technique for embedding watermark bits into digital audio signals. The proposed method is based on the patchwork algorithm on the wavelet domain and does not need the original audio signal in the watermark detection. It uses the wavelet transform generated by the low-pass analysis filter h_n whose length is 2 and $h_0 = h_1 = 1$ to account for a fast synchronization between watermark embedding and detection parts. Several simulation results show that the proposed method is robust against various signal manipulations such as MPEG/Audio layer 3 compression and time scale modification.

keywords: Wavelets, synchronization, patchwork, time scale modification.

*Hong Oh Kim is with Division of Applied Mathematics, KAIST, 373-1 Kusong-Dong, Yusong-Gu, Taejon, 305-701, Korea, hkim@ftn.kaist.ac.kr. This work is supported partly by KOSEF 98-0701-0301-5.

†Bae Keun Lee is with Division of Applied Mathematics, KAIST, 373-1 Kusong-Dong, Yusong-Gu, Taejon, 305-701, Korea, bklee@amath.kaist.ac.kr.

‡Nam-Yong Lee is with Division of Applied Mathematics, KAIST, 373-1 Kusong-Dong, Yusong-Gu, Taejon, 305-701, Korea, nylee@amath.kaist.ac.kr. This work is supported partly by Brain Korea 21 Program.
Corresponding Author

1 Introduction

Digital data have several advantages. They can be shared by multiple users, distributed over network, and managed for long period time without any damage. In contrast to those advantages, the copyright protection problem arises, since unauthorized copying and distribution of digital data are simplified, too. With widespread use of Internet and proliferation of digital contents (audio, image, and video, etc) distribution, the copyright protection of digital contents is becoming more important and difficult.

Conventional encryption algorithms permit only authorized users to access encrypted digital data. Once such data are decrypted, however, there is no way in prohibiting its illegal copying and distribution. The digital watermarking is intended to complement the weakness of the encryption algorithm in protecting the intellectual rights on digital data. The digital watermarking hides an information to host data in a sense that the added one does not destroy the basic appearance of the data to person using it. The added information is called digital watermark and used to carry information about the copyright holder of data, copy control information, or individualized information about the license holder in order to track illicit copies, etc.

Recently, various image watermarking techniques have been introduced. There is a large interest in audio watermarking techniques which are largely stimulated by the rapid progress in audio compression algorithm and wide use of Internet for compressed music distribution over the recent years.

The audio watermark should be inaudible, statistically unnoticeable to prevent unauthorized removal, robust to intentional signal processing attacks such as compression, filtering, resampling, noise adding, digital-to-analog/analog-to-digital conversion, etc, and self-clocking for ease of detection in the presence of time scale modification attack.

The audio watermarking can be classified into temporal watermarking and spectral watermarking, based on the domain where watermarks are embedded. Temporal watermarking [1] [2] [3] hides watermarks directly into digital audio signals in the time domain, and spectral watermarking methods [4] [5] [6] [7] first transform given audio signals, where FFT(Fast Fourier Transform), DCT(Discrete Cosine Transform), and DWT(Discrete Wavelet Transform), etc, are commonly used as the underlying transform, and hides watermarks in the transform domain.

It is known that the temporal audio watermarking is relatively easy to implement and requires less computing resources, as compared with the spectral watermarking. On the other hand, however, the temporal watermarking is weaker than the spectral watermarking against general signal processing attacks such as audio compression and filtering, etc.

The spectral audio watermarking applies certain frequency transform, such as FFT, DCT, and DWT, etc, to the data block of the audio signal, and hides the watermark information into the transformed data block. In audio watermarking, it is impossible to have the same information on locations of data blocks, where the frequency transform is applied to, between watermark embedding and detection parts due to the time scale modification attack. Therefore, to be robust against the time scale modification attack, the spectral audio watermarking must use the fast algorithm that quickly finds the data block where the watermark bit is actually embedded.

The patchwork algorithm [2] artificially modifies the difference(we call this *patch value* in this work) between estimated sums of samples in two randomly chosen and prescribed index subsets. Thus the modified patch value is many deviation away from expected. The artificial modification can be detected, with a high probability, by comparing the observed patch value with the expected one. The temporal patchwork algorithm, however, is very weak to the time scale modification attack, since the patchwork algorithm depends on two prescribed index subsets. Moreover, audio compression, filtering, and resampling also hurt the performance of the temporal patchwork algorithm. On the other hand, the patchwork on the frequency domain is quite robust to audio compression, filtering, and resampling, etc. Furthermore, since the transformed data in the frequency domain have little changes by the relatively small time scale modification, the spectral patchwork algorithm is strong to the time scale modification attack. But, this is true only for the correct data block, where the frequency transform and the the patchwork algorithm are applied, or just few samples departed data blocks form the correct one. Therefore, to be robust against the time scale modification attack, the spectral patchwork algorithm must use a fast watermark detection method to check data blocks as many as possible within given time limit.

In this work we suggest to use the patchwork algorithm on the piecewise constant DWT(see Section 2 for the definition) to overcome the described difficulties in the audio watermarking. The proposed method achieves the robustness by using the patchwork algorithm on the frequency domain and the fast synchronization between watermark embedding and detection parts by using the fast watermark detection algorithm that is sufficiently fast to check every possible data block(which might be the one where the watermark bit is embedded) in a reasonable time.

As compared with the standard spectral patchwork algorithm, which uses first the frequency transform to the data block and then detects the watermark in the transformed data block(See, e.g., a DCT-based patchwork algorithm [7]), the proposed method does not need the DWT of the data block at any time in the watermark detection, and is very fast. Such improvement in the speed of the watermark detection comes from the fact that the proposed method examines the abrupt change in the difference of consecutive patch values rather than patch value itself.

The main benefit of using the piecewise constant DWT in the spectral patchwork algorithm is the fact that the difference between the wavelet coefficient of one sample shifted data block and that of original one is computable directly and quickly from the audio data in the time domain. Therefore, in examining the abrupt change in the difference of consecutive patch values, the proposed method does not require the DWT of the data block at any time. For FFT and DCT, there are similar fast algorithms in updating the transformed data of the one sample shifted data block from those of the original data block. However, the difference between them is neither directly nor quickly computable from the audio data in the time domain. This means that at least one time FFT or DCT of the data block is required for the watermark detection in the FFT or DCT-based method. Obviously, this reduces the speed of the watermark detection.

We conducted watermark embedding and detection experiment for test audio signals to show the performance of the proposed method. With a sufficient redundancy on the watermark bits, the proposed method perfectly detects 50 watermark bits that are embedded into audio signals of 33 second length. It is also shown through the experiment that the proposed method is robust to various signal processing manipulations such as MPEG/Audio layer 3 audio compression and time scale modification, etc., as long as the quality of audio is not severely damaged.

The rest of the paper is organized as follows. In Section 2 we explain the watermark embedding method by the patchwork algorithm on the piecewise constant DWT. In Section 3 we propose the watermark detection algorithm. In Section 4 we explain the simulation result of the proposed method. Finally, we give concluding remarks in Section 5.

2 Patchwork on the DWT Domain

The proposed method of this paper uses the patchwork algorithm [2] on the DWT domain, where the underlying DWT is generated by the low-pass analysis filter h_n whose length is 2 and $h_0 = h_1 = 1$. In this section we present necessary concept of the DWT for the presentation of this paper. See [8] [9] [10] for more detailed theories and applications of DWT.

The basic idea in the DWT for an one dimensional signal is the following. A signal is decomposed into two parts, high frequencies and low frequencies. The discontinuity components of the signal are largely confined to the high frequency part. The low frequency part is decomposed again into two parts of high and low frequencies. The number of decompositions in above process is usually determined by the application and the length of the original signal. The data obtained from the above decomposition are called the DWT coefficients. Moreover, from these DWT coefficients, the original signal can be reconstructed. This reconstruction

process is called the inverse DWT.

To be specific, the (biorthogonal) DWT [8] is defined by analysis filters $(h_n), (g_n)$ and synthesis filters $(\tilde{h}_n), (\tilde{g}_n)$, which satisfy

$$\sum_n h_n \tilde{h}_{n+2k} = 2\delta_{k,0},$$

$$g_n = (-1)^{n+1}\tilde{h}_{-n+1},$$

and

$$\tilde{g}_n = (-1)^{n+1} h_{-n+1}.$$

Here $(h_n), (\tilde{h}_n)$ and $(g_n), (\tilde{g}_n)$ are called low-pass filters and high-pass filters, respectively. For the orthogonal DWT [9] we have $h_n = \tilde{h}_n$.

For given discrete data $(x_n), n = 0, 1, \ldots, 2^m - 1$, (In this paper, to simplify our presentation, we always assume that the input data of the DWT is of length 2^m for some positive integer m), let $C_n^0 = x_n$. The DWT of f is obtained by successively applying

$$\begin{array}{rcl} C_j^{k+1} & = & \sum_n h_{n-2j} C_n^k \\ D_j^{k+1} & = & \sum_n g_{n-2j} C_n^k \end{array} \tag{1}$$

to $C^k, k = 0, 1, \ldots, k_0 - 1$. That is, the DWT maps C^0 to $D^1, D^2, \ldots, D^{k_0}, C^{k_0}$ for some positive integer k_0. On the other hand, the inverse DWT reconstructs C^0 from $D^1, D^2, \ldots, D^{k_0}$, C^{k_0} by successively using

$$C_j^k = \frac{1}{2}\sum_n \tilde{h}_{j-2n} C_n^{k+1} + \frac{1}{2}\sum_n \tilde{g}_{j-2n} D_n^{k+1}. \tag{2}$$

Here we assume that for each k the sequences C_j^k and D_j^k are periodic with the period 2^{m-k}. With this periodicity assumption, the DWT coefficients C_j^k and D_j^k are computed in (1) and (2).

Several $\tilde{h}_n$(as low-pass synthesis filter) generate the DWT with h_n(as low-pass analysis filter) whose length is 2 and $h_0 = h_1 = 1$. We write them in z-notation, i.e., $\sum_n \tilde{h}_n z^n$, as follows.

$$\begin{array}{c} 1 + z \\ \\ -\frac{1}{8}z^{-2} + \frac{1}{8}z^{-1} + 1 + z + \frac{1}{8}z^2 - \frac{1}{8}z^3 \\ \\ \frac{3}{128}z^{-4} - \frac{3}{128}z^{-3} - \frac{11}{64}z^{-2} + \frac{11}{64}z^{-1} + 1 \\ +z + \frac{11}{64}z^2 - \frac{11}{64}z^3 - \frac{3}{128}z^4 + \frac{3}{128}z^5 \end{array} \tag{3}$$

For these filters, we have

$$C_j^{k-1} = \sum_{n=j2^{k-1}}^{(j+1)2^{k-1}-1} f_n \tag{4}$$

and

$$D_j^k = \sum_n g_{n-2j} C_n^{k-1}. \tag{5}$$

Let L be the length of the filter $\tilde{h}_n$. Suppose that $\hat{\mathcal{B}}$ is the one sample shifted data block of

$$\mathcal{B} = (f_s, f_{s+1}, \ldots, f_{s+2^m-1})$$

from audio signal (f_n), that is,

$$\hat{\mathcal{B}} = (f_{s+1}, f_{s+2}, \ldots, f_{s+2^m}).$$

Notice that the wavelet coefficient $\hat{D}_j^{k_0}$ of $\hat{\mathcal{B}}$ can be quickly computable from the wavelet coefficient $D_j^{k_0}$ of $\mathcal{B}$ by

$$\hat{D}_j^{k_0} = D_j^{k_0} + \sum_n g_{n-2j}(f_{s+(n+1)2^{k_0-1}} - f_{s+n2^{k_0-1}}) \tag{6}$$

as long as

$$j \in \{n_0, n_0+1, \ldots, 2^{m-k_0} - 1 - n_0\}, \quad n_0 = (L-2)/4.$$

The lengths of $\tilde{h}_n$ in (3) are 2, 6, 10, respectively. Thus for those filters, n_0 is a nonnegative integer. The restriction generated by n_0 guarantees that the computation of wavelet coefficients of $\hat{\mathcal{B}}$ and $\mathcal{B}$ is not affected by the periodicity assumption imposed on given data blocks. In this paper, we are mainly interested in the DWT generated from above h_n and $\tilde{h}_n$, and call it *piecewise constant* DWT.

Similar results as (6) hold for FFT and DCT coefficients. The n-th FFT coefficient $\hat{F}_n$ of $\hat{\mathcal{B}}$ can be quickly computable from the n-th FFT coefficient F_n of $\mathcal{B}$ by

$$\hat{F}_n = (F_n - f_s + f_{s+2^m}) e^{-2\pi i n/2^m}. \tag{7}$$

Let C_n and S_n be the n-th DCT coefficient and DST(Discrete Sine Transform) of the data block $\mathcal{B}$, that is,

$$C_n = \sum_{k=0}^{2^m-1} f_{s+k}\ cos\left(\frac{\pi n(k+1/2)}{2^m}\right)$$

and

$$S_n = \sum_{k=1}^{2^m-1} f_{s+k}\ sin\left(\frac{\pi nk}{2^m}\right).$$

Then one can update the n-th DCT and DST coefficients $\hat{C}_n$ and $\hat{S}_n$ of $\hat{\mathcal{B}}$ from C_n and S_n obtained from $\mathcal{B}$ by using following relations:

$$\hat{C}_n = C_n + 2sin\left(\frac{\pi n}{2^{m+1}}\right) S_n + (f_{s+2^m}(-1)^n - f_s)\ cos\left(\frac{\pi n}{2^{m+1}}\right) \tag{8}$$

and

$$\hat{S}_n = S_n - 2sin\left(\frac{\pi n}{2^{m+1}}\right) \hat{C}_n \tag{9}$$

In this work we use binary watermarks (w_i)(i.e., $w_i = 0$ or 1). The proposed method hides one watermark bit to one data block of the audio signal. The audio watermark can be viewed as a signal that is transmitted through a communication channel, which is the watermarked audio signal in this case. Attacks and unintentional audio signal distortion are thus regarded as noise that the watermark must be immune to it. To have a safe communication between the embedding and the detection of watermarks, we give a redundancy on the binary watermark bits by repeating them *locally*. We also add several bits in front of the watermark bits to locate the point where watermark bits begin to be embedded. We call such added bits as the *synchronization bits*. For example, with the local redundancy rate 3 and the synchronization bits 10101011 of length 8, we change the original watermark bits as

$$w_0w_1w_2\ldots \longrightarrow 10101011w_0w_0w_0w_1w_1w_1w_2w_2w_2\ldots.$$

With abuse of terminology, we call these watermark bits.

The proposed watermark embedding method proceeds as follows. We first take a data block

$$\mathcal{B}_i = (f_{s_i}, f_{s_i+1}, \ldots, f_{s_i+2^{m-1}-1}), \quad s_{i+1} = s_i + 2^m,$$

from the audio signal (f_n), and then apply the piecewise constant DWT to $\mathcal{B}_i$ to have D^1, D^2, $\ldots, D^{k_0}, C^{k_0}$ for some positive integer k_0. The first point s_0 is chosen (arbitrarily) as a sufficiently large number, for example, $s_0 > 100,000$, not to embed the watermark bits to the silent region in the beginning part of audio signals.

We apply the patchwork algorithm [2] to the coarsest wavelet coefficients D^{k_0}. The patchwork in the DWT domain proceeds as follows. Define the patch value $\mathcal{P}_N$ by

$$\mathcal{P}_N = \sum_{\mu\in\mathcal{I}} D^{k_0}_\mu - \sum_{\nu\in\mathcal{J}} D^{k_0}_\nu,$$

where disjoint index subsets $\mathcal{I}$ and $\mathcal{J}$ are randomly chosen from $\{n_0, n_0+1, \ldots, 2^{m-k_0}-1-n_0\}$, $n_0 = (L-2)/4$. We artificially modifies $\mathcal{P}_N$ to add a statistical pattern in a way that the modified $\mathcal{P}_N$ is many deviation away from expected. To be specific, we modify some wavelet coefficients in D^{k_0} as

$$D^{k_0}_\mu \to D^{k_0}_\mu + \delta, \quad D^{k_0}_\nu \to D^{k_0}_\nu - \delta, \quad \text{if} \quad x_i = 1, \tag{10}$$

and

$$D^{k_0}_\mu \to D^{k_0}_\mu - \delta, \quad D^{k_0}_\nu \to D^{k_0}_\nu + \delta, \quad \text{if} \quad x_i = 0, \tag{11}$$

for $\mu \in \mathcal{I}$ and $\nu \in \mathcal{J}$, where x_i is the watermark bit to be embedded into the data block $\mathcal{B}_i$. Here we suggest to use same index subsets $\mathcal{I}$ and $\mathcal{J}$ for the audio blocks that are used in embedding of the synchronization bits and different index subsets for the true watermark bits for security purpose.

Finally, we apply the inverse piecewise constant DWT to the wavelet coefficients $D^1, D^2, \ldots, \tilde{D}^{k_0}, C^{k_0}$ to have the watermarked data block $\tilde{\mathcal{B}}_i$, where $\tilde{D}^{k_0}$ is the modified wavelet coefficients in the previous step. We repeat the described steps to the next data block until no watermark bits are left for embedding.

Notice that $\mathcal{P}_N$ of the watermarked data block $\tilde{\mathcal{B}}_i$ follows either

$$\mathcal{P}_N \sim \mathcal{N}(2N\delta, \sigma^2), \quad \text{if } x_i = 1, \tag{12}$$

or

$$\mathcal{P}_N \sim \mathcal{N}(-2N\delta, \sigma^2), \quad \text{if } x_i = 0. \tag{13}$$

Thus by comparing $\mathcal{P}_N$ with $N\delta$, with a high probability, we can accurately estimate the watermark bit x_i embedded in the watermarked data block $\tilde{\mathcal{B}}_i$ without knowing the original data block $\mathcal{B}_i$.

We summarize described procedures as follows.

Watermark Embedding

$i = 0$;
while(watermark bits are left for embedding){
  DWT: $\mathcal{B}_i \to D^1, D^2, \ldots, D^{k_0}, C^{k_0}$;
  Patchwork: $D^{k_0} \to \tilde{D}^{k_0}$;
  IDWT: $D^1, D^2, \ldots, \tilde{D}^{k_0}, C^{k_0} \to \tilde{B}_i$;
  $s_{i+1} = s_i + 2^m$;
  $i = i + 1$;
}

3 Watermark Detection

In audio watermarking, it is impossible to have the same information on starting points $s_0, s_1, s_2 \ldots$ of data blocks in both embedding and detection parts of watermarks due to the time speed modification attack. The watermark detection of the proposed method need neither the information about those starting points nor the original audio signal.

Let

$$\mathcal{P}_N^t = \sum_{\mu \in \mathcal{I}} D_\mu^{k_0} - \sum_{\nu \in \mathcal{J}} D_\nu^{k_0},$$

where $D_j^{k_0}$ are the wavelet coefficients computed from the data block

$$\mathcal{A}_t = (f_t, f_{t+1}, \ldots, f_{t+2^m-1}).$$

We define the difference $\triangle_N^t$ by

$$\triangle_N^t = \mathcal{P}_N^{t+1} - \mathcal{P}_N^t.$$

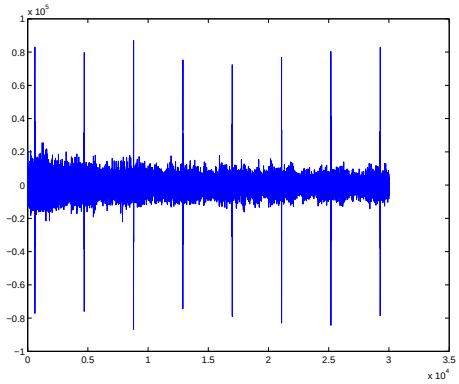

Figure 1: S_N^t of the watermarked and unattacked audio signal.

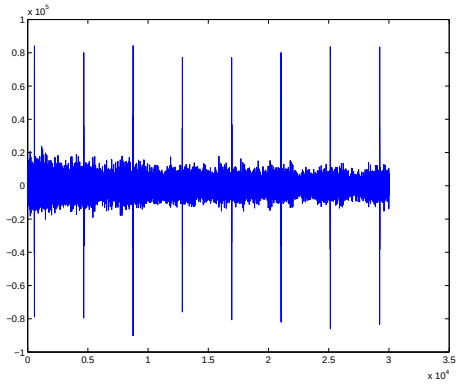

Figure 2: $\triangle_N^t$ of the watermarked and unattacked audio

Figure 1 and Figure 2 show $\mathcal{P}_N^t$ and $\triangle_N^t$, respectively, of a watermarked audio signal. As we can see from Figure 1 and Figure 2, the watermarking effect(the peaks in graphs) in $\triangle_N^t$ is equally noticeable to that in $\mathcal{P}_N^t$. Moreover, the computation of $\triangle_N^t$ is much faster than that of $\mathcal{P}_N^t$. This is the reason why we use $\triangle_t$ instead of $\mathcal{P}_N^t$ in the watermark detection.

For the fast computation of $\triangle_N^t$, notice that

$$\begin{aligned}
\triangle_N^t &= \mathcal{P}_N^{t+1} - \mathcal{P}_N^t \\
&= \textstyle\sum_{\mu\in\mathcal{I}}(\hat{D}_\mu^{k_0} - D_\mu^{k_0}) - \sum_{\nu\in\mathcal{J}}(\hat{D}_\nu^{k_0} - D_\nu^{k_0}) \\
&= \textstyle\sum_{\mu\in\mathcal{I}}\sum_p g_{p-2\mu}(f_{t+(p+1)2^{k_0-1}} - f_{t+p2^{k_0-1}}) \\
&- \textstyle\sum_{\nu\in\mathcal{J}}\sum_p g_{p-2\nu}(f_{t+(p+1)2^{k_0-1}} - f_{t+p2^{k_0-1}}),
\end{aligned}$$

where $\hat{D}^{k_0}$ and D^{k_0} are wavelet coefficients of $\mathcal{A}_{t+1}$ and $\mathcal{A}_t$, respectively. For the last equality

in above equations we have used (6). Thus the computation of $\triangle_N^t$ requires

$$\begin{array}{ll} 4NL-2N+1 & \text{additions and} \\ 2NL & \text{multiplications.} \end{array} \tag{14}$$

The standard DWT needs

$$\begin{array}{ll} (2^m+2^{m-1}+\ldots+2^{k_0+1})L & \text{additions and} \\ (2^m+2^{m-1}+\ldots+2^{k_0+1})L & \text{multiplcations} \end{array} \tag{15}$$

in computing wavelet coefficients D^{k_0} only. In most audio watermarking applications, the number in (14) is much smaller than that in (15).

We can further reduce the operation count (14). For the piecewise constant DWT generated by the low-pass synthesis filter with the length $L = 10$, the third one in (3), notice that

$$g_{-4} = g_{-3} = -g_4 = -g_5 = -\frac{3}{128}$$

and

$$g_{-2} = g_{-1} = -g_2 = -g_3 = \frac{11}{64}.$$

Thus we have

$$\begin{aligned} &\textstyle\sum_p g_{p-2\mu}(f_{t+(p+1)2^{k_0-1}} - f_{t+p2^{k_0-1}}) \\ &= (-\tfrac{3}{128})(f_{t+(2\mu-2)2^{k_0-1}} - f_{t+(2\mu-4)2^{k_0-1}} \\ &\qquad + f_{t+(2\mu+4)2^{k_0-1}} - f_{t+(2\mu+6)2^{k_0-1}}) \\ &\quad +(\tfrac{11}{64})(f_{t+2\mu 2^{k_0-1}} - f_{t+(2\mu-2)2^{k_0-1}} \\ &\qquad + f_{t+(2\mu+2)2^{k_0-1}} - f_{t+(2\mu+4)2^{k_0-1}}) \\ &\quad + f_{t+(2\mu+2)2^{k_0-1}} - 2f_{t+(2\mu+1)2^{k_0-1}} + f_{t+2\mu 2^{k_0-1}}. \end{aligned}$$

Therefore the total operation count in computing $\triangle_N^t$ is

$$\begin{array}{ll} 20N+1 & \text{additions and} \\ 6N & \text{multiplications} \end{array}$$

for the piecewise constant DWT generated by the low-pass synthesis filter with the length $L = 10$. A similar reduction in the operation count holds for other piecewise constant DWT.

Notice that by using (6) the difference between the wavelet coefficient of one sample shifted data block and that of original one can be computable directly and quickly from the audio data in the time domain. Thus the described fast algorithm in the computation of $\triangle_N^t$ is only available for the piecewise constant DWT.

Obviously, it is better to use $\mathcal{P}_N^t$ for FFT or DCT-based patchwork algorithm. The computation of $\mathcal{P}_N^{t+1}$ can be quickly updated from that of $\mathcal{P}_N^t$ by using (7) for FFT-based method

and (8) and (9) for DCT-based method. But, unlike the proposed method, which uses the patchwork algorithm in the piecewise constant DWT domain, FFT or DCT-based patchwork algorithm needs to compute the transform of the data block at least one time. In practice, we cannot scan every possible data block by forwarding sample by sample to detect the watermark bit. Therefore FFT or DCT-based patchwork algorithm needs to compute the transform of the data block many times. Certainly, this reduces the speed in the watermark detection.

We suggest to use the following criterion for the watermark detection:

Detection Criterion: For fixed $\beta > 0$,

- if
$$\triangle_N^t > \beta N\delta \quad \text{and} \qquad \triangle_N^{t+s} < 0, \quad s = 1 \text{ or } 2$$
then 1 is detected in the data block $\mathcal{A}_t$.

- if
$$\triangle_N^t < -\beta N\delta \quad \text{and} \qquad \triangle_N^{t+s} > 0, \quad s = 1 \text{ or } 2$$
then 0 is detected in the data block $\mathcal{A}_t$.

- if previous two cases are not satisfied, then no watermark bit is detected in the data block $\mathcal{A}_t$.

The watermark detection consists of two parts: One for finding the synchronization bits that are used to locate the starting point of the true watermark bits and the other for detecting the watermark bit embedded in the next data block once the watermark detection for the current data block is done.

The watermark detection in the proposed method proceeds as follows. We first take a data block $\mathcal{A}_{t_i}, i = 0, 1, \ldots$, and compute

$$\triangle_N^{t_i+n}, \quad -2^m/10 \le n < 2^m/10.$$

Here the first point t_0 is chosen from sufficiently small numbers, not to have any watermark bits before it.

We use the Detection Criterion to $\triangle_N^t$. If one watermark bit is detected at n, then we go to the next data block $\mathcal{A}_{t_{i+1}}$ with

$$t_{i+1} = t_i + n + 2^m.$$

On the other hand, if no watermark bit is detected, then we go to the next data block $\mathcal{A}_{i+1}$ with

$$t_{i+1} = t_i + 2^m/10 + r_i 2^m,$$

where r_i is the randomly chosen number from $[0, 1]$. By using this randomized approach, we can find quickly one of synchronization bits with a high probability.

Once we find a watermark bit in $\mathcal{A}_{t_i}$, we might expect the next watermark bit in the data block $\mathcal{A}_{t_{i+1}}$ with the new starting point $t_{i+1} = t_i + 2^m$. Due to the time scale modification attack, however, the next watermark may not be found in $\mathcal{A}_{t_{i+1}}$. To make the proposed method to be robust against the time speed modification attack, it is desirable to search for the watermark bit in data blocks $\mathcal{A}_{t_{i+1}+j}, -2^m/10 \le j \le 2^m/10$. We search for the watermark bit with the following order:

$$j = 0, -1, 1, -2, 2, \ldots.$$

If we detect the watermark bit in $\mathcal{A}_{t_{i+1}+j}$ for some j, then we stop the search and go to the next data block with the new starting point

$$t_{i+2} = t_{i+1} + j + 2^m.$$

If no watermark is detected in $\mathcal{A}_{t_{i+1}+j}$ for all j, $-2^m/10 \le j \le 2^m/10$, then we do not report the watermark bit for the data block $\mathcal{A}_{t_{i+1}}$ and go to the next data block $\mathcal{A}_{t_{i+2}}$ with

$$t_{i+2} = t_{i+1} + 2^m.$$

We repeat the described steps until all watermark bits are assumed to be detected.

With the piecewise constant DWT generated by the low-pass synthesis filter with the length $L = 10$, the worst case in the detection of one watermark bit needs only

$2^m(20N + 1)/5$	additions and
$2^m\, 6N/5$	multiplications.

Thus the worst case computation with the piecewise constant DWT, for small N and L, is comparable to the computation with FFT, DCT, or non-piecewise constant DWT with the case when no time scale modification is applied to the data block and that data block is exactly the one where the watermark bit is embedded.

We summarize the described watermark detection method as follows.

Watermark Detection

$i = 0$;
while(1){
  for$(n = -2^m/10; n < 2^m/10; n = n + 1)$*{*
    Compute $\triangle_N^{t_i+n}$;
    Detection Criterion on $\triangle_N^{t_i+n}$;
    if(one watermark bit is detected){

$t_{i+1} = t_i + n + 2^m$;
goto label;
}
}
$t_{i+1} = t_i + 2^m/10 + r_i 2^m$;
$i = i + 1$;
label: *break*;
}

while(watermark bits are left to be detected){
for($j = 0; j < 2^m/10; j = j + 1$){
Compute $\triangle_N^{t_{i+1}+j}$;
Detection Criterion on $\triangle_N^{t_{i+1}+j}$;
if(one watermark bit is detected){
$t_{i+2} = t_{i+1} + j + 2^m$;
$i = i + 1$;
break;
}
Compute $\triangle_N^{t_{i+1}-(j+1)}$;
Detection Criterion on $\triangle_N^{t_{i+1}-(j+1)}$;
if(one watermark bit is detected){
$t_{i+2} = t_{i+1} - (j + 1) + 2^m$;
$i = i + 1$;
break;
}
}
if(no watermark bit is detected in $\mathcal{A}_{t_{i+1}+j}$ *for all* j*)*{
$t_{i+2} = t_{i+1} + 2^m$;
$i = i + 1$;
}
}

4 Simulations

We conducted watermark embedding and detection experiment for 4 test audio signals, which are sampled at 44.1kHz with 16-bit depth.

We embedded one watermark bit to the data block of 4096 samples($m = 12$). We used the

piecewise constant DWT generated by the filter $\tilde{h}_n$(see (3)) of length 10($L = 10$) and apply the DWT to the data block with $k_0 = 3$. For the patchwork step, we used $N = 20(N = |\mathcal{I}| = |\mathcal{J}|)$ and $\delta = 4000$. We also used $\beta = 0.7$ in the Detection Criterion.

We embedded 50 watermark bits to audio signals of 33 second length. In order to have the 100% success rate in the watermark detection, we used the 16 synchronization bits and the local redundancy rate 3. In most audio signals, the failure in the watermark detection often happens consecutively. Such case is rare, but, if it happens, it is not easily avoidable just by increasing the local redundancy rate. To overcome this problem, we suggest to repeat the true watermark part(excluding the synchronization bits) three times(i.e., the global redundancy rate = 3).

Table 1 shows the result of the simulation. The first and second columns show the music names used in the simulation and the number of detected bits out of 450(50 watermark bits $\times$ local redundancy rate 3 $\times$ global redundancy rate 3) bits that are embedded in the watermarked audio data as the true watermark bits, respectively. As we can see, the proposed method did not provide the perfect detection, but with the local redundancy rate 3 and the global redundancy rate 3, we had 100% success rate in the watermark detection for all four audio signals used in the simulation.

Table 1.

Success rate of the watermark detection. The number denotes the detected bits from the 450 embedded watermark bits. MP3 = MPEG/Audio layer 3 compression, TSM(+4%) = Time Scale Modification with 4% increment.

	No attack	MP3	TSM(+4%)
Light music	393	211	154
Ballad music	389	159	247
Pop music	418	265	343
Chamber music	420	351	293

In Table 1 the third columns shows numbers of detected bits out of 450 bits as the second column, but after applying the MPEG/Audio layer 3 compression algorithm with bit rate 128kbps to watermarked audio signals. As compared with numbers in the second column(the case with no attack), the raw success rates are decreased, but they are sufficiently high to have 100% success rate with the described redundancy. At a first glance, the success rates seems to be too low, because some of them are even below 50%. In general, the probability of the false watermark detection is much smaller than that of the failed watermark detection. The failed

watermark detection does not affect the decision. Thus if the correct detection rate is much higher than the false detection rate, then we can have 100% success rate even with a small raw success rate.

The fourth column in Table 1 shows the raw success rates after applying the pitch-invariant 4% time speed increment. Again, as compared with the raw success rates in the case with no attack, numbers are decreased, but they are sufficiently high to have 100% success rates with the described redundancy.

In the case when the pitch-invariant 4% time speed decrement, we did not have 100% success rate even with the described redundancy. The pitch-invariant time speed decrement totally removes some part of audio signal while preserving the pitch. Thus for the data block where the missing part is relatively large, we can not detect the watermark bit. The pitch-invariant time speed increment, however, adds some signal to some part of the original audio signal. Thus locations, where watermarks bits are detected, might be irregular (see Figure 4), but we can detect all watermark bits successfully. The failure caused by the pitch-invariant time speed decrement can be overcome by using a bigger data block length at the expense of the watermark embedding capacity.

By using bigger δ or N in the patchwork part, one can have the same success rate in the watermark detection with smaller local and global redundancy rates. In most cases, however, such a result comes with possible degradation of quality in the watermarked audio signal.

Figure 3, Figure 4, and Figure 5 show plots of $\triangle_N^t$ of the watermarked audio signal after applying the MPEG/Audio layer 3 compression algorithm with bit rate 128kbps, the pitch-invariant 4% time speed increment and decrement, respectively. The peaks caused by the watermark embedding are still noticeable after applying described attacks.

The proposed method spent 8.2 seconds for the watermark embedding and 2.9 seconds for the watermark detection in the unattacked audio signals with the computer with Pentium III 866MHz cpu and 512 MB ram. The watermark detection in attacked audio signals spent 5.5 seconds. The watermark detection with the full scanning, i.e., the case which computes $\triangle_N^t$ for all t, spent 48.9 seconds for the audio signal with the 33 second length.

5 Conclusion

In this paper we proposed a novel audio watermarking technique, which uses the patchwork algorithm on the piecewise constant DWT domain. The proposed method provides a fast synchronization between the watermark embedding and detection parts without original audio signals. The proposed method is very robust against the MPEG/Audio layer 3 audio compres-

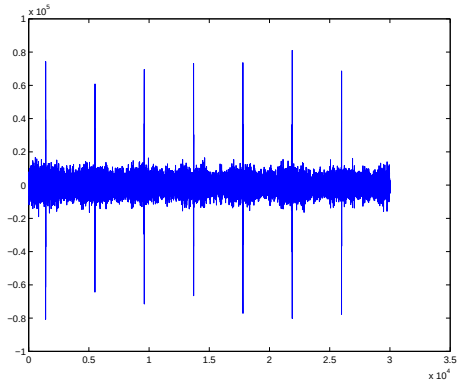

Figure 3: $\triangle_N^t$ of the watermarked audio signal after applying the MPEG/Audio layer 3 compression with bit rate 128kbps.

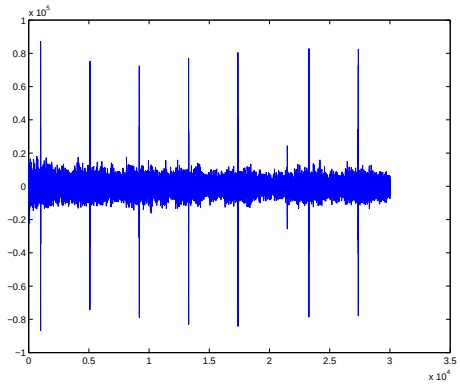

Figure 4: $\triangle_N^t$ of the watermarked audio signal after applying the pitch-invariant 4% time speed increment.

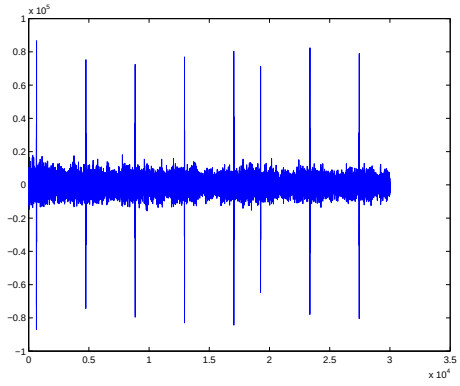

Figure 5: $\triangle_N^t$ of the watermarked audio signal after applying the pitch-invariant 4% time speed decrement.

sion algorithm and the time scale modification, as long as the quality of the audio signal is not too severely damaged.

By using the multiscale structure in wavelet coefficients, we can hide the watermark bits in different purpose. For example, we can apply the patchwork algorithm to the coarsest wavelet coefficients for the synchronization purpose as we did in this paper, and use various other watermarking methods in less coarser wavelet coefficients for the information hiding purpose.

As compared with the proposed method, the FFT or DCT-based patchwork algorithm to is slower because of the occasional computation of $\mathcal{P}_N^t$. On the other hand, the updating step in the FFT or DCT-based patchwork algorithm is as fast as that in the proposed method. Thus if the time required to the occasional computation of $\mathcal{P}_N^t$ is bearable, then the FFT or DCT-based patchwork algorithm equipped with the fast updating algorithms based on (7) or (8) and (9) can be a useful robust audio watermarking method.

References

[1] P. Bassia, I. Pitas, and N. Nikolaidis,"Robust audio watermarking in the time domain", IEEE Transactions on Multimedia, Vol. 3, June 2001, pp. 232 -241.

[2] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, "Techniques for data hiding", IBM Systems Journal, Vol. 35, 1996, pp. 313–336.

[3] C. Xu, J. Wu, and Q. Sun, "A robust digital audio watermarking technique", Fifth International Symposium on Signal Processing and Its Applications, Brisbane, Australia, 22-25 Aug., 1999.

[4] L. Boney, A.H. Tewfik, and K.N. Hamdy, "Digital watermarks for audio signals", In International Conference on Multimedia Computing and Systems, pp. 473-480, IEEE, Hiroshima, Japan, 17–23 Jun. 1996.

[5] I. J. Cox, J. Kilian, T. Leighton, and T. Shamoon, "Secure Spread Spectrum Watermarking for Multimedia", IEEE Transaction on Image Processing, Vol. 6, No. 12, 1997, pp. 1673–1687.

[6] D. Kirovski and H. Malvar, " Robust spread-spectrum audio watermarking", 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 3, 2001, pp. 1345-1348.

[7] I.-K. Yeo, and H. J. Kim,"Modified Patchwork Algorithm: a novel audio watermarking scheme", International Conference on nformation Technology: Coding and Computing, 2001, pp. 237 -242.

[8] A.Cohen, I. Daubechies, and J.-C. Feauveau, "Biorthogonal Bases of Compactly Supported Wavelets", Comm. Pure Appl. Math., Vol 45, 1992, pp. 485-560.

[9] I. Daubechies "Orthonormal bases of compactly supported wavelets", Comm. Pure and Appl. Math. 1988 v.41, pp. 909–996.

[10] I. Daubechies, " Ten Lectures on Wavelets", SIAM, Philadelphia, 1992.