J Inf Process Syst, Vol.15, No.2, pp.271~287, April 2019
https://doi.org/10.3745/JIPS.03.0112

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

A Synchronization Scheme Based on Moving Average for Robust Audio Watermarking

Jinquan Zhang* and Bin Han*

Abstract

The synchronization scheme based on moving average is robust and suitable for the same rule to be adopted in embedding watermark and synchronization code, but the imperceptibility and search efficiency is seldom reported. The study aims to improve the original scheme for robust audio watermarking. Firstly, the survival of the algorithm from desynchronization attacks is improved. Secondly, the scheme is improved in inaudibility. Objective difference grade (ODG) of the marked audio is significantly changed. Thirdly, the imperceptibility of the scheme is analyzed and the derived result is close to experimental result. Fourthly, the selection of parameters is optimized based on experimental data. Fifthly, the search efficiency of the scheme is compared with those of other synchronization code schemes. The experimental results show that the proposed watermarking scheme allows the high audio quality and is robust to common attacks such as additive white Gaussian noise, requantization, resampling, low-pass filtering, random cropping, MP3 compression, jitter attack, and time scale modification. Moreover, the algorithm has the high search efficiency and low false alarm rate.

Keywords

Audio Watermarking, Moving Average, Robust Watermarking, Synchronization Code

1. Introduction

So far, many audio watermarking methods have been developed [1]. For the copyright protection of digital audio works, a scheme is usually required to resist common signal processing operations such as additive noise, resampling, MP3 compression, and low-pass filtering. Low-pass filtering can produce different time delay due to different cut-off frequencies. MP3 compression may add about 1,000 0-value samples in the front of the audio. For different audio works or different bit rates, the number of the added 0 is not exactly the same. Time delay or 0-value samples affect the search efficiency of the exact location for some algorithms. Additionally, desynchronization operations, such as randomly cutting, inserting or replacing audio content and jitter attack, are also dangerous to robust watermarking algorithms. In order to ensure the accuracy of watermark extraction, the synchronization measure is adopted to determine the extraction range or position.

In these algorithms which can resist the desynchronization operation, various techniques were adopted: histogram shape [2], log coordinate mapping feature [3], statistics average value [4-6], logarithmic discrete

Manuscript received July 21, 2017; first revision September 11, 2017; accepted October 23, 2017.
Corresponding Author: Jinquan Zhang (zhjq@cuit.edu.cn)
* School of Cybersecurity, Chengdu University of Information Technology, Chengdu, China (zhjq@cuit.edu.cn, binhan_cn@163.com)

cosine transform [7], low frequency coefficient of discrete wavelet transform (DWT) [8], and others [9-11]. In these algorithms, the adoption of synchronization code is wide concerned.

In those schemes utilizing synchronization codes, a synchronization code is embedded, followed by a message. When the watermarked audio is attacked by desynchronized operations such as randomly cutting and jitter attack, exhaustive search is often performed to obtain the synchronization code. In order to search a synchronization code, the maximal number of samples required to traverse is the sum of samples to embed synchronization code and message once. The more execution times of the extraction algorithm means the higher false alarm rate because the algorithm is required to match the synchronization information from more detected bits.

In the previous study [10], the synchronization code is the front 12 bits of the 13-bit Barker code. The 16-bit/sample audio clip is used to embed message by changing the lower 13 bits of samples and one bit synchronization code is embedded into one sample. In the experimental results, the robustness of the scheme is strong and the search efficiency of the synchronization code is high. The embedding strength is large. When one-bit message is embedded, only one sample is adjusted and the distortion of the audio waveform is obvious. Moreover, there is "click" at the position of the synchronization code, as mentioned in the previous study [12].

Wu et al. [8] embedded the synchronization code in the low frequency coefficient of DWT domain with quantization index modulation [13]. The synchronization code was a 63-bit m-sequence. In the extraction phase, as long as more than 42 bits were the same, it was considered that the synchronization code is detected. Although the computational cost during searching for synchronization code could be reduced according to the method described in the paper, the efficiency was still relatively low because it adopted sample-by-sample searching.

Lie and Chang [14] proposed a watermarking scheme. The embedding method of watermark was the same to the synchronization code. The algorithm exploited differential amplitude relations in every three consecutive sections of samples to represent one-bit information. In order to improve the quality of the watermarked audio, the algorithm smoothed the boundary of sections, and the psychoacoustic model was applied to control the amount of watermark disturbances below the masking thresholds. In order to reduce the change in the amplitude and improve the robustness of the algorithm, about 1,000 samples were required for embedding one-bit message. The extraction algorithm needs to perform about $l\times(n_1+n_2)/20$ to find one synchronization mark, where l is the number of samples of three consecutive sections; n_1 is the number of bits of the synchronization code; n_2 is the number of bits of the watermark. In experiments, l=1020 and n_1=20. The synchronization search efficiency is also relatively low.

Wang and his colleagues [4,5] adopted the same algorithm to embed synchronization code. The algorithm can be considered as an improvement from the Wang's method [10]. Wang et al. [4] employed Barker code to locate the position where watermark was embedded. In their scheme, the synchronization code was embedded into the statistical average value of multiple consecutive audio samples. In their experiments, the number of consecutive samples was 5. So, the efficiency of searching synchronization code was high. For the number of consecutive samples was 5, the embedding strength was set to be 0.2 in order to improve the robustness. Therefore, the distortion of the waveform was relatively large, thus leading to audible noise [12]. In order to improve the embedding algorithm while maintaining the robustness, it is suggested to embed one-bit message into more consecutive samples and reduce the embedding strength. However, the synchronization search efficiency will decrease and false alarm rate

will increase. With this technique, the effects of embedding synchronization marks on objective difference grade (ODG) and bit error rate (BER) were explored [15].

In the previous synchronization method [12], a given bits of sequence or its inverse was embedded into consecutive samples of the audio signal. It was a novel time domain synchronization technique. So, the execution of the scheme was fast enough for real-time applications. In their experiments, the number of consecutive samples was 4. The number was so small that the scheme was not robust enough for common signal processing operation.

In the previous synchronization scheme [6], synchronization bits were embedded by adjusting the positive and negative of the mean value of multiple continuous samples. In fact, in order to avoid the interferences on the auditory quality of watermarked audio, the mean value was close to 0, indicating that the number of consecutive samples should be sufficient. In their experiments, the number of continuous samples was 484 and the synchronization code was 16 bits. The sample-by-sample searching was used to extract the synchronization code phase, so the searching efficiency is low.

We argue that, for a synchronization scheme, robustness, inaudibility and search efficiency are three important factors. The search efficiency of their synchronization scheme is high [10,12], but the noise introduced by embedding synchronization code [10] is obvious and the robustness [12] is relatively poor. The robustness of the synchronization scheme [6,14] is strong, but the searching efficiency is relatively low.

An audio signal is a one-dimensional signal. Therefore, it is important to embed synchronizing information in time domain. At the same time, embedding information in time domain tends to have the higher embedding capacity and detection efficiency. In this paper, we improve our previous study [16]. The proposed synchronization scheme based on moving average for robust audio watermarking achieves a compromise among robustness, inaudibility, and search efficiency.

This paper is organized as follows. In Section 2, we introduce the concept of moving average in audio signal. Section 3 presents our synchronization code embedding and extraction strategies. Section 4 shares the improvement in our rule. In Section 5, we present experimental results and performance analysis results of our scheme. Finally, conclusions are drawn in Section 6.

2. Moving Average in Audio Signal

2.1 Definition of Moving Average in Audio Signal

Moving average (MA) is a term in statistics. It is also called moving mean (MM) or rolling mean. A moving average can be viewed as an example of a low-pass filter used in signal processing.

Assuming the number of samples of an audio clip X is L, and samples' amplitude are denoted as x_1, x_2, …, x_L. After choosing an integer b, the moving average M_B is defined as

$$M_{B_i} = \frac{x_i + x_{i+1} + \cdots + x_{i+b-1}}{b}, i \in (1, L - b + 1). \quad (1)$$

We give an example to illustrate moving average in Fig. 1. In the example, "Sample" is the audio waveform and M_A and M_B are respectively, the moving averages for a=16 and b=26.

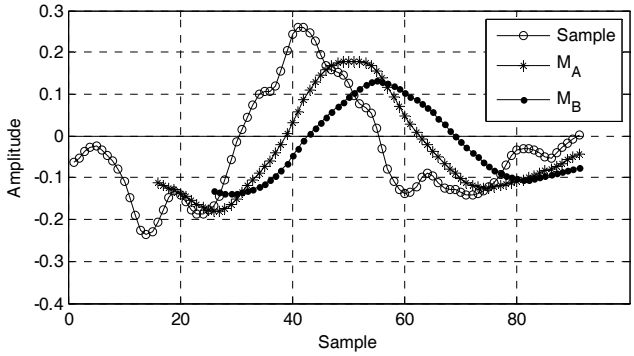

Fig. 1. Example of moving average.

2.2 Cross of Two Moving Average

Two different integers (a and b, $a<b$) are chosen. Then M_A and M_B are obtained according to Eq. (1). For a certain M_{B_i} in M_B, there is a corresponding $M_{A_{i+b-a}}$ in M_A in time axis. If

$$\left(M_{B_i} - M_{A_{i+b-a}}\right)\left(M_{B_{i+1}} - M_{A_{i+b-a+1}}\right) \leq 0, i \in (1, L-b+1), \tag{2}$$

We believe that there is a cross between M_A and M_B.

2.3 Rapid Calculation of Moving Average

The following method is used to obtain the MA of an audio segment quickly, especially in digital signal process.

Assume an audio signal $X=(x_1, x_2, \ldots, x_L)$. For a given integer b>0,

Set

$$v_i = x_i + x_{i+1} + \cdots + x_{i+b-1} \tag{3}$$

Then

$$M_{B_i} = v_i/b \tag{4}$$

Now, we compute $\mathrm{M}_{B_{i+1}}$. In fact, compared with M_{B_i}, in the computation of $\mathrm{M}_{B_{i+1}}$, the i^{th} sample is excluded and the $(i+b)$th sample is included.

Then

$$v_{i+1} = v_i - x_i + x_{i+b} \tag{5}$$

$$M_{B_{i+1}} = v_{i+1}/b \tag{6}$$

The method can reduce the computational load dramatically.

In fact,

$$M_{B_{i+1}} = M_{B_i} + (x_{i+b} - x_i)/b. \tag{7}$$

3. Synchronization Scheme

In the proposed scheme, two proper positive integers are chosen to compute the moving average. Then, the synchronization bits are embedded at the crosses of the two moving averages with the quantization index modulation method [13]. The proposed synchronization scheme is described in the following sections.

3.1 Choice of Synchronization Code

In audio watermarking schemes, the synchronization code is widely used to locate the watermark. In the previous study [10], the front of 12 bits of 13-bit Barker code was chosen as the synchronization code. The m-sequence with 63 bits was also used as the synchronization code [8]. In previous studies [4,6,12], the length of the synchronization code was 16 bits and consisted of the 13-bit Barker code concatenating 3-bit Barker code. Chaotic sequence was also adopted in previous studies [17,18], but the length was not given. In addition, the length of 20-bit synchronization code was also employed [14], but the type of synchronization code was not described.

It is assumed that the length of synchronization code is l bits. If t bits are correct, it is considered that the synchronization code is detected. Then the false alarm rate of an algorithm can be expressed as [8]:

$$\mathrm{P}_{FA} = \frac{1}{2^l}\sum_{k=t}^{l} C_l^k. \tag{8}$$

C_l^k means the number of k-combinations from n elements.

When exhaustive search is used to detect the synchronization code, the extraction algorithm usually employs the method that one bit message is extracted when moving forward one sample. Therefore, the search efficiency is low and the number of extracted bits is large. Moreover, the false alarm rate is high. For a synchronization code algorithm, the higher search efficiency and the lower false alarm rate are still important except robustness.

Compared with the watermark information, the synchronization code is often very short. In the previous study [4], the style, length, and the probability of "0" and "1" of synchronization code are taken into account. The length of synchronization code is especially important. The longer the synchronization code is, the lower the false alarm is.

When synchronization code is detected, if $t=l$ is required in Eq. (8), the false alarm rate is only related to the length of the synchronization code, regardless of the type of the synchronization code. That is, $\mathrm{P}_{FA}=1/2^l$. If $t<l$ is allowed, the false alarm rate is closely related to the type of synchronization code. Since the Barker code and m-sequence have low sidelobes, they are both suitable for synchronization code.

(a) **m-sequence**: An m-sequence is a periodic sequence that can be generated by a linear feedback shift register. For linear feedback systems with k shift registers, if the period of its output sequence is 2^k-1, the sequence is called m-sequence.

Assuming two m-sequences a_n and b_n with N bits: a_n, $b_n \in \{-1,1\}$, $n \in [1,N]$. The cross-correlation functions of a_n and b_n are provided as follows:

$$R_{a,b}(j) = \sum_{n=1}^{N} a_n\, b_{n-j}. \tag{9}$$

If $a_n \in \{-1,1\}$, $n \in [1, N]$ is a m-sequence, $N=2^k-1$. The autocorrelation function of m-sequence has the following properties:

$$R_{a,a}(j) = \begin{cases} N, j = 0 \\ -1, 0 < j < N \end{cases}. \tag{10}$$

According to Eq. (8), false alarm rate of the algorithm obtained when l=31, t>28 is less than that obtained when l=16, t=16. False alarm rate of the algorithm is about 10^{-5} when l=31, t>28 or l=16, t=16.

(b) **Barker code**: Barker code is a binary code group, which is not a periodic sequence. The value of each symbol is +1 or -1. The autocorrelation function of an n-bit Barker code is provided as follows:

$$R_{x,x}(j) = \begin{cases} n, j = 0 \\ \pm 1, j \neq 0 \end{cases}. \tag{11}$$

Only nine Barker sequences are known and the longest length n is 13 bits. It is known from the above analysis that the shorter the synchronization code is, the greater the false alarm rate is. In order to reduce the false alarm rate, the Barker code (1 1 1 1 1 -1 -1 1 1 -1 1 -1 1) and (1 1 -1) are concatenated as the synchronization code [4,5].

3.2 Embedding Process

Two different integers (a and b, $a<b$) are selected. M_A and M_B are computed according to Eq. (1). The process of embedding a message is described as follows.

It is assumed that the length of the synchronization code S is l bits and that the embedding strength is s.

Sometimes two crosses are very close to each other. In order to improve the robustness of the algorithm, for the first bit of the synchronization code, the distance from the first sample to be counted to the current cross, which the first bit of the synchronization code will be embedded into, is greater than b and excludes other crosses. If the above conditions are not satisfied, the first bit of the synchronization code may not be detected. For other bits of the synchronization code, the distance from the former cross, which one bit message has been embedded into, to the current cross, which one bit message will be embedded into, is also greater than b and other crosses are allowed to be included.

The embedding process is described as follows.

(1) Set $d[-1]$=3s/4, $d[1]$=s/4, cnt=0, i=1.

(2) Search a cross of M_A and M_B until the end of the audio clip.
do while $\left(M_{B_i} - M_{A_{i+b-a}}\right)\left(M_{B_{i+1}} - M_{A_{i+b-a+1}}\right) > 0${
i=i+1,
cnt = cnt +1,
}

(3) If cnt <=b and the first bit of the synchronization code will be embedded, then cnt =0, and go to step (2).

(4) If cnt < =b, then go to step (2). If cnt >b, then go to step (5) to embed one bit message.

(5) Assume $u = M_{B_{i+1}}$. For the k^{th} bit of synchronization code $S(k)$, the embedding rule is provided as follows:

$$u' = \begin{cases} round\left(\frac{u+d[1]}{s}\right) \times s - d[1], S(k) = 1 \\ round\left(\frac{u+d[-1]}{s}\right) \times s - d[-1], S(k) = -1 \end{cases}. \quad (12)$$

round() denotes the rounding function.

Set $d = u' - u$. Notice that the i^{th} point of the sequence M_B corresponds to the $(i+b\text{-}1)^{th}$ sample of the audio in time axis. The amplitude of each sample from the former cross to the sample x_{i+b} is added to d. For the first bit of the synchronization code, the starting position is the first sample to be counted.

(6) Set *cnt* =0. Go to step (2) to embed the next bit.

In the embedding rule, since the amplitude of the original audio is directly modified and the parameter *cnt* is larger than *b*, the crossed position of M_A and M_B will not change, as described in Section 5.3.

3.3 Detecting Process

Assuming the integers *a* and *b* and the embedding strength *s*, the synchronization code is obtained for the detection algorithm. Then M_A and M_B are computed according to Eq. (1). The detecting process is provided as follows:

(1) Set *d*[-1]=3s/4, *d*[1]=s/4, *cnt*=0, *i*=1.

(2) Search a cross of M_A and M_B until the end of the audio clip.
do while $\left(M_{B_i} - M_{A_{i+b-a}}\right)\left(M_{B_{i+1}} - M_{A_{i+b-a+1}}\right) > 0${
i=*i*+1,
cnt = *cnt* +1,
}

(3) If *cnt* < =b, go to Step (2). If not, go to Step (4) to extract one bit message.

(4) Assume $u' = M'_{B_{i+1}}$. The detecting rule is provided as follows:

$$m(k) = \begin{cases} 1, u' - \left\lfloor \frac{u'}{s} \right\rfloor \times s \geq s/2 \\ -1, u' - \left\lfloor \frac{u'}{s} \right\rfloor \times s < s/2 \end{cases}. \quad (13)$$

(5) Set *cnt* =0. Go to Step (2) to search the next cross.

Then the cross-correlation between the synchronization code and the extracted information is calculated as follows:

$$r(k) = \sum_{i=1}^{l} S(i)m(k-l+i), k \geq l. \quad (14)$$

For the synchronization code *S*, since the length of *S* is *l* bits and the value of each bit is 1 or -1, the autocorrelation is *l*.

Searching in $r(x)$, the position of the value *l* is the position of the synchronization code.

Usually, if the distance of two *l* is too close or too far, one of them will be ignored because the marked audio is destroyed by cutting and adding or there is a false synchronization.

From the embedding and detecting phases, we can see that, for the first bit of the synchronization code, it is required that the distance from the first sample to be counted to the current cross excludes other crosses in order to ensure that the first bit can be detected with the higher probability.

4. Improvements in the Embedding Scheme

In the previously proposed scheme [14], audible noise was introduced into the marked audio when the embedding strength was large. An idea was previously presented to solve this problem [14]. However, we do not think the idea is a good one. Fox example, it is not proper when the amplitude of two adjacent segments simultaneously increases or decreases. So, the core of the solution is to make the changes of two adjacent segments as smooth as possible. In our scheme, the solution is described as follows.

It is assumed that c_1 is the change of the former segment and that c_2 is that of the current segment. T_N samples in the front of current segment will be involved in the improvement. For the first bit of synchronization code, c_1=0. For the sake of simplicity, it is assumed that the variation of T_N samples is linear. That is to say, it conforms to the linear function y=kx+c, x=0, 1, 2, …, T_N. When $x = 0$, the sample is the last one of the former segment and the distortion of the amplitude is c_1. When $x = \mathrm{T}_N$, the distortion of the sample is c_2. So,

$$\begin{cases} c_1 = c \\ c_2 = kT_N + c \end{cases} \tag{15}$$

We get the linear function is $y = (c_2 - c_1)x/T_N + c_1, x = 0,1,2, \ldots, T_N$.

Although the improved method has little effect on the signal-to-noise ratio of the marked audio, the ODG value is significantly improved and the robustness of the algorithm decreases insignificantly, as described in Sections 5.1, 5.4, and 5.5.

5. Performance Analysis and Experiment

In the experiments, we test our algorithm with different audio clips including pop music, light music, march music, country music, and Blues music. The experimental results are similar for all audio files. We report the results with three audio clips including the pop music clip, light music clip, and Blues music clip. They are in WAV format, mono, 16 bits/sample, 16 seconds, and 44.1 kHz sampling frequency. The synchronization code is made up of 13-bit Barker code 1 1 1 1 1 -1 -1 1 1 -1 1 -1 1 concatenating 3-bit Barker code 1 1 -1.

5.1 Analysis of Signal-to-Noise Ratio

In audio watermarking schemes, the signal-to-noise ratio (SNR) is a difference indicator between the watermarked and the original audio. The definition of SNR is shown as follows:

$$\mathrm{SNR} = 10\lg\frac{\sum_{i=1}^{L} x_i^2}{\sum_{i=1}^{L}(x_i - x_i^*)^2}, \tag{16}$$

where x_i and x_i^* are respectively, the original and marked audio signals.

Assuming the embedding strength is s, for a given audio clip, $\sum_{i=1}^{L} x_i^2$ is a constant. According to the embedding algorithm, since the number of samples is large, we presume that the amplitude change of samples follows a uniform distribution. That is to say, $(X^*\text{-}X) \sim U(\text{-s/2,s/2})$, where X^* is the watermarked audio. Therefore, the expected value $E(X^*\text{-}X)$=0, the variance $D(X^*\text{-}X)$=s^2/12.

$$E((X^*\text{-}X)^2)= D(X^*\text{-}X)+(E(X^*\text{-}X))^2= s^2/12. \quad (17)$$

$$\sum_{i=1}^{L}(x_i - x_i^*)^2 \approx \sum_{i=1}^{L}\frac{s^2}{12} = \frac{s^2 L}{12} \quad (18)$$

So,

$$\text{SNR} = 10\lg\frac{\sum_{i=1}^{L}x_i^2}{\sum_{i=1}^{L}(x_i - x_i^*)^2} \approx 10\lg\frac{\sum_{i=1}^{L}x_i^2}{\frac{s^2 L}{12}} = 10\lg\frac{12\sum_{i=1}^{L}x_i^2}{s^2 L} = 10\lg\frac{12\sum_{i=1}^{L}x_i^2}{L} - 20\lg s \quad (19)$$

From Eq. (19), the SNR is mainly affected by the embedding strength. Taking the pop music as an example, set a=26, b=40, the synchronization code was embedded into the audio clip, repeatedly. The variations of experimental SNR and derived results with the embedding strength are provided in Table 1. When other audios are used, the similar result also occurs.

Table 1. SNR evaluation

s	0.001	0.002	0.004	0.006	0.008	0.01	0.012	0.014	0.016	0.018	0.02
SNR	54.035	47.933	41.968	38.439	35.98	33.962	32.372	31.155	29.941	28.952	28.027
#DSNR	53.949	47.928	41.908	38.386	35.887	33.949	32.365	31.026	29.867	28.844	27.928

s=the embedding strength of the synchronization code, #DSNR=SNR of the derived result according to Eq. (19).

SNR is an important indicator to evaluate a watermarking algorithm. Generally, the larger embedding strength means the less SNR, the higher robustness of watermarked audio, and the more obvious noise. For SNR does not take the characteristics of human auditory into account, perceptual evaluation of audio quality (PEAQ), an assessment tool recommended by ITU BS1387, is also used in objective evaluation tests.

5.2 Adaptive Choice of Parameters

From Section 5.1, we know that the SNR is related to the embedding strength of the watermark algorithm. We often set the embedding strength s$\in$(0.01,0.02). Then, the value of a is respectively, equal to 22, 26, 30, 36, 40, 46, and 50, the value of b is incremented from a+10 step by 2 until 2a. Common signal processing operations are used to test the robustness of the marked audio. At last, an acceptable way is provided as follows.

Assuming that the number of zero-crossings of M_{10} is $Z_{M_{10}}$, where M_{10} is obtained according to Eq. (1). L is the number of the samples of the original audio clip. Set $num=L/Z_{M_{10}}$. The integer b in embedding phase in Section 3 is slightly less than num to resist jitter attack and time scale modification attack.

For most styles of audio clips, usually, a is about 2b/3.

After the parameter b is chosen, the embedding strength can be decided according to the following method.

The effects of some common signal processing operations on the waveform of an audio file are observed in experiments.

A light music clip is chosen for low-pass filtering. A 6th-order Butterworth filter with the cut-off frequency of 6 kHz is used. After the processed audio is shifted to the left by four sampling periods, the amplitude of its samples is subtracted by that of the original audio. Then, the change of each sample is obtained.

In the experiment, 32 consecutive samples are chosen randomly. The waveform of the original audio

is shown in Fig. 2(a) and the waveform of the processed audio is shown in Fig. 2(b). Time delay is 4 sampling periods. Although the waveforms in Fig. 2(a) and 2(b) are almost the same, the biggest difference between them exceeds 0.01, as shown in Fig. 2(c). Here, the processed audio clip is shifted to the left by 4 sampling periods. To the whole audio clip, the distribution of the differences between their moving averages is shown in Fig. 2(d). When the same filter is used, the lower the cutoff frequency of the low-pass filter is, the greater the introduced noise is.

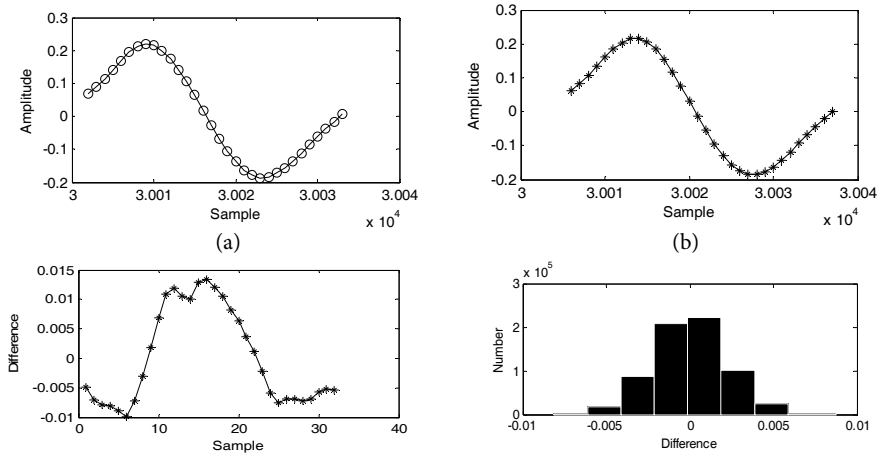

Fig. 2. Effects of the low-pass filtering on audio signal: (a) waveform of the original audio, (b) waveform of the processed audio signal, (c) difference between the processed and original audio signals, and (d) distribution histogram of the difference of moving average.

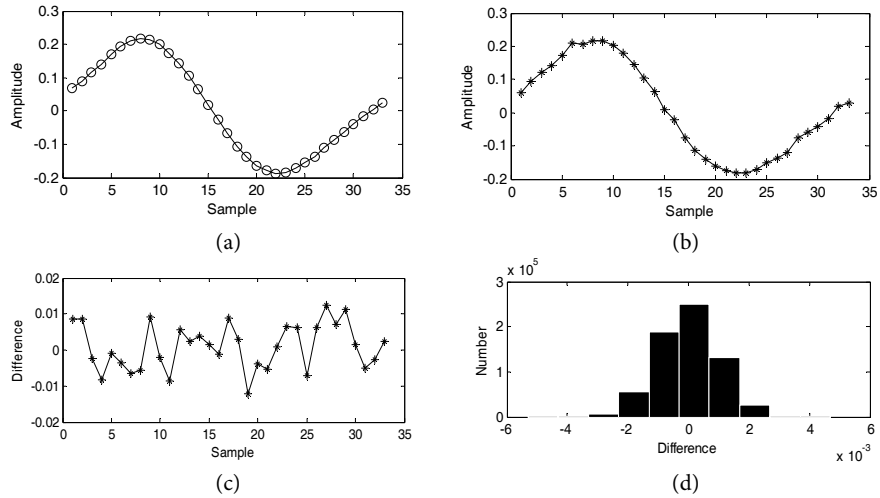

Fig. 3. Effects of AGWN on audio signal: (a) waveform of the original audio, (b) waveform of the processed audio signal, (c) difference between the processed and original audio signals, and (d) distribution histogram of the difference of moving average. Note: Data presented in Fig. 3(c) are not derived by subtracting the data in Fig. 3(a) from the data in Fig. 3(b). The data in Fig. 3(b) are different in each execution of AWGN under the same SNR. For the same reason, the data in Fig. 3(d) are in each execution of AWGN under the same SNR.

When additive Gaussian white noise (AGWN) is used, the similar experiment is performed with the same audio. In the experiment, the SNR of noise is 45 dB. We choose the same 32 consecutive samples in the previous experiment. The waveform of the original audio is shown in Fig. 3(a) and the waveform of the processed audio is shown in Fig. 3(b). Although the waveforms in Fig. 3(a) and 3(b) are almost the same, the biggest difference between them exceeds 0.01, as shown in Fig. 3(c). Although the biggest difference exceeds 0.01, in the whole audio clip, the distribution of differences among their moving averages is shown in Fig. 3(d).

For other signal operations, such as re-quantization, resampling, and MP3 compression, the similar experiment can be made to analyze the differences. The embedding strength is chosen according to the distribution of differences. At the same time, if watermark and synchronization code are repeatedly embedded, the correct rate of extracted bits is only required to meet a given value.

5.3 Analysis of the Influences of Embedding Marks on Cross Location

As described in Section 2.2, two different integers a and $b(a<b)$ are chosen. For a certain M_{B_i} in M_B, there is a corresponding $M_{A_{i+b-a}}$ in M_A in time axis. If

$$\left(M_{B_i} - M_{A_{i+b-a}}\right)\left(M_{B_{i+1}} - M_{A_{i+b-a+1}}\right) \le 0, i \in (1, L-b+1),$$

there is a cross between M_A and M_B.

Here, we assume $M_{B_i} < M_{A_{i+b-a}}$, then, $M_{B_{i+1}} > M_{A_{i+b-a+1}}$. According to Eq. (1), before the mark bit is embedded,

$$M_{A_{i+b-a}} = (x_{i+b-a} + x_{i+b-a+1} + \cdots + x_{i+b-1})/a, \tag{20}$$

$$M_{B_i} = (x_i + x_{i+1} + \cdots + x_{i+b-1})/b. \tag{21}$$

After the mark bit is embedded, according to the embedding rule, we get

$$M'_{A_{i+b-a}} = \frac{(x_{i+b-a}+d)+(x_{i+b-a+1}+d)+\cdots+(x_{i+b-1}+d)}{a} = M_{A_{i+b-a}} + d \tag{22}$$

$$M'_{B_i} = \frac{(x_i+d)+(x_{i+1}+d)+\cdots+(x_{i+b-1}+d)}{b} = M_{B_i} + d \tag{23}$$

So,

$$M'_{B_i} - M'_{A_{i+b-a}} = M_{B_i} - M_{A_{i+b-a}}. \tag{24}$$

$$M'_{B_{i+1}} - M'_{A_{i+b-a+1}} = M_{B_{i+1}} - M_{A_{i+b-a+1}}. \tag{25}$$

So, the crossed positions of M_A and M_B are not changed.

5.4 Imperceptibility Comparison Before and After Improvement

An inaudibility comparison of the synchronization code method before and after improvement is provided in Table 2. In the experiment, the Barker code with 16 bits is used as synchronization code, a pseudo-random sequence with 128 bits is used as the watermark message. The watermark and the

synchronization code are embedded repeatedly with the same rules. For the improved algorithm, T_N=5.

In Table 2, EA indicates that the improved algorithm is not used for watermark and synchronization code. EB means that the improved algorithm is adopted for embedding watermark, but not for synchronization code. EC denotes that the improved algorithm is employed for embedding both watermark and synchronization code.

For light music, a=16, b=24, the embedding strength of the synchronization code is 0.012, and that of the watermark is 0.011. For pop music, a=26, b=40, the embedding strength of the synchronization code is 0.016, and that of the watermark is 0.015. The experimental result is shown in Table 2.

Table 2. Imperceptibility comparison before and after improvement

		EA	EB	EC
Light music	#TE	127	127	127
	#TD	126	126	118
	SNR (dB)	33.6	33.9	33.6
	ODG	-1.59	-1.30	-0.98
	MOS	4.5	4.8	5.0
Pop music	#TE	72	72	72
	#TD	72	72	69
	SNR (dB)	30.4	30.5	30.4
	ODG	-2.96	-2.25	-0.86
	MOS	3.2	3.8	5.0
Blues music	#TE	58	58	58
	#TD	58	58	54
	SNR (dB)	27.4	27.5	27.5
	ODG	-1.30	-1.00	-0.55
	MOS	4.2	4.9	5.0

Usually, the embedding strength of watermark is less than that of the synchronization code. The larger the embedding strength is, the more robust the synchronization code is. It is a compromise between robustness and imperceptibility.

The second line (#TE): Times of embedding synchronization code in the marked signal.

The third line (#TD): Times of detecting synchronization code in the marked signal (without attack).

The fourth line (SNR): SNR according to Eq. (16).

The fifth line (ODG): objective evaluation tests with PEAQ.

The sixth line (mean opinion score [MOS]): the test is performed by a team composed of 10 audiences.

As shown in Table 2, the synchronization code is still not completely detected without any attack because when the synchronization code is detected, two correlated crosses with a value of 16 are too close, as described in Section 3.3.

The marked audio has the better quality after improvement due to the boundary processing. Without the improved algorithm, the difference of the samples between the marked audio and the original audio has a jump at the boundary, as shown in Fig. 4(a). When the improved algorithm is used, the changes in the samples at the boundary are relatively smooth, as shown in Fig. 4(b).

For Blues music and country music, which have a lot of high frequency components, the effect of the improved algorithm is still obvious. For Blues music, a=30, b=46, the embedding strength of the synchronization code is 0.02, and that of the watermark is 0.02. The experimental result is shown in Table 2.

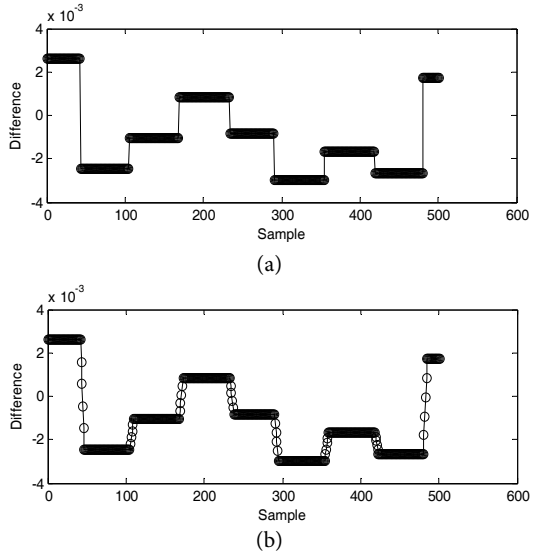

Fig. 4. Effects of the improved algorithm on the audio samples: (a) difference of the samples between the marked audio and the original audio when the improved algorithm is not used and (b) difference of the samples between the marked audio and the original audio when the improved algorithm is used.

5.5 Robustness Comparison Before and After Improvement

A robustness comparison of the synchronization code method for signal operations before and after improvement is provided in Table 3. In the experiments, T_N=5, whereas the values of other parameters, such as a, b, and embedding strength, are the same, as described in Section 5.4. The experimental results are shown in Table 3.

(1) Additive white Gaussian noise: white Gaussian noise is added to the marked signal.
(2) Re-quantization: The 16-bit marked audio signals are re-quantized down to 8 bits/sample and then back to 16 bits/sample.
(3) Resampling: The marked signal, originally sampled at 44.1 kHz, is re-sampled at 22.05 kHz/11.025 kHz, and then restored back by sampling again at 44.1 kHz.
(4) Low-pass filtering: A sixth-order Butterworth filter with the cut-off frequency of 10 kHz/8 kHz/4 kHz is used.
(5) Cropping: Segments of 10% are removed from the marked audio signal randomly.
(6) MP3 compression 128 kbps/96 kbps/80 kbps: The MPEG-1 layer-3 compression is applied. The marked audio signal is compressed at the bit rate of 128 kbps/96 kbps/80 kbps and then decompressed back to the WAVE format.
(7) Jitter attack: Randomly remove one sample out of every 1,000/500/200 samples from the marked signals.

(8) Time scale modification (TSM): The marked signals are scaled in time domain, where the scaling factors are ±3%, ±7%, and ±10%.

#TD: Times of detecting the synchronization code with the improved scheme.

#TDorg: Times of detecting the synchronization code with the original scheme.

As shown in Table 3, the robustness of the improved algorithm is slightly decreased. Combined the robustness of the marked audio with the audibility, the improved algorithm is better than the original one.

Table 3. Robustness comparison before and after improvement

Attack	Pop music		Light music	
	#TD	#TDorg	#TD	#TDorg
No attack	69	72	114	126
AWGN				
55 dB	63	65	98	100
45 dB	51	52	45	48
Requantization	66	67	114	119
Resampling				
11,025 Hz	63	67	110	114
22,050 Hz	68	71	114	122
Low-pass				
10 kHz	58	55	41	42
8 kHz	64	67	100	104
4 kHz	54	52	37	40
Cropping (10%)	60	64	101	111
MP3				
80 kbps	38	31	10	11
96 kbps	47	44	29	23
128 kbps	52	54	48	49
Jitter				
1/1000	65	63	90	100
1/500	59	55	75	77
1/200	43	43	36	41
TSM				
-3%	59	61	86	94
-7%	48	51	60	81
-10%	45	44	56	57
3%	68	71	102	110
7%	64	67	77	84
10%	61	64	66	73

5.6 Robustness Comparison with Other Schemes

The experimental data for light music and pop music were compared with the previous results [12]. Compared with the previous study [12], more samples are involved in embedding one-bit message in our scheme. As shown in Table 4, the proposed algorithm is robust to common attacks. However, the synchronization scheme in the previous study [12] is fast in the embedding phase and it is applicable to real-time scene.

Table 4. Robustness comparison with other schemes

Attack	Pop music		Light music		Ref. [12]	
	#TD	# PD (%)	#TD	# PD (%)	#TD	#PD (%)
No attack	69	96	118	93	117	100
AWGN (55 dB)	63	88	99	78	NR	NR
Re-quantization	66	92	114	90	33	28
Re-sampling						
22,050 Hz	68	94	116	91	26	22
11,025 Hz	63	88	110	87	0	0
Cropping (10%)	60	83	104	82	105	90
Low-pass						
10 kHz	58	81	41	32	7	6
8 kHz	64	89	102	80	NR	NR
4 kHz	54	75	37	29	NR	NR
MP3						
128 kbps	52	72	43	34	60	51
96 kbps	47	65	28	22	36	31

#TD=times of detecting synchronization code correctly, #PD=proportion of detecting synchronization code correctly, NR=the experimental data of the selected scheme were not reported.

5.7 Search Efficiency Comparison with Other Algorithm

The search efficiency comparison with other algorithms is provided in Table 5. It is assumed that the lengths of the synchronization code and watermark are respectively n_1 bits and n_2 bits and that the numbers of the samples involved in embedding one-bit synchronization code and watermark are respectively l_1 and l_2. In order to search the synchronization code, exhaustive search is adopted in all algorithms. That is, the maximum number of the samples for searching synchronization code is $l_1 \times n_1 + l_2 \times n_2$.

Table 5. Comparison of search efficiency

Ref.	#ENE	#VP	#ND
[14]	$l_1 \times (n_1+n_2)$	$l_1=l_2=1020$	102,000
[12]	$l_1 \times n_1 + l_2 \times n_2$	$l_1=4$, $l_2=512$	43,072
[6]	$l_1 \times (n_1+n_2)$	$l_1=l_2=484$	48,400
Our study	$l_2 \times n_2/l_1+n_1$	$l_1=l_2<100$	100

In Table 5, the expressions in column 2 (#ENE) are the maximum execution times of detection algorithm required to find a synchronization code. The values of the parameters l_1 and l_2 given in the previous studies are shown in column 3 (#VP). The numbers of the samples for embedding one-bit synchronization code and for embedding one-bit watermark are the same in literature [6,14] and our scheme. That is, $l_1=l_2$. For the convenience of comparison, $n_1=16$ and $n_2=84$ for all algorithms. According to the data in Column 3, the execution times of the detection algorithm are shown in column 4 (#ND).

In [14], the position of the synchronization code can be coarsely computed. Then, the search speed is markedly improved by almost 20 times without decreasing the accuracy of alignment. It is still about 5,000.

By comparison, we can see that our algorithm has the higher search efficiency. Less information is extracted to search for the synchronization code, so the false alarm rate is decreased.

6. Conclusion

For moving average is robust to common signal processing operations, our scheme embeds synchronization code into the cross of two moving average sequences. We discussed the imperceptibility of the scheme and compared it with other algorithms in search efficiency. The experimental results show that the proposed watermarking scheme maintains the high audio quality and is robust to common attacks. Simultaneously, the algorithm has the high search efficiency and low false alarm rate.

Acknowledgement

We thank Prof. Hongxia Wang for valuable suggestions. This study was supported by the Scientific Research Foundation of CUIT (No. KYTZ201420), and Scientific Research Projects of Department of Education in Sichuan Province (No. 16ZA0221).

References

[1] G. Hua, J. Huang, Y. Q. Shi, J. Goh, and V. L. Thing, "Twenty years of digital audio watermarking: a comprehensive review," *Signal Processing*, vol. 128, pp. 222-242, 2016.

[2] S. Xiang and J. Huang, "Histogram-based audio watermarking against time-scale modification and cropping attacks," *IEEE Transactions on Multimedia*, vol. 9, no. 7, pp. 1357-1372, 2007.

[3] X. Kang, R. Yang, and J. Huang, "Geometric invariant audio watermarking based on an LCM feature," *IEEE Transactions on Multimedia*, vol. 13, no. 2, pp. 181-190, 2011.

[4] X. Y. Wang, P. P. Niu, and H. Y. Yang, "A robust digital audio watermarking based on statistics characteristics," *Pattern Recognition*, vol. 42, no. 11, pp. 3057-3064, 2009.

[5] X. Y. Wang and H. Zhao, "A novel synchronization invariant audio watermarking scheme based on DWT and DCT," *IEEE Transactions on Signal Processing*, vol. 54, no. 12, pp. 4835-4840, 2006.

[6] V. Bhat, I. Sengupta, and A. Das, "An adaptive audio watermarking based on the singular value decomposition in the wavelet domain," *Digital Signal Processing*, vol. 20, no. 6, pp. 1547-1558, 2010.

[7] Y. Xiang, I. Natgunanathan, S. Guo, W. Zhou, and S. Nahavandi, "Patchwork-based audio watermarking method robust to de-synchronization attacks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 9, pp. 1413-1423, 2014.

[8] S. Wu, J. Huang, D. Huang, and Y. Q. Shi, "Efficiently self-synchronized audio watermarking for assured audio data transmission," *IEEE Transactions on Broadcasting*, vol. 51, no. 1, pp. 69-76, 2005.

[9] M. Arnold, X. M. Chen, P. Baum, U. Gries, and G. Doerr, "A phase-based audio watermarking system robust to acoustic path propagation," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 3, pp. 411-425, 2014.

[10] J. Huang, Y. Wang, and Y. Q. Shi, "A blind audio watermarking algorithm with self-synchronization," in *Proceedings of 2002 IEEE International Symposium on Circuits and Systems*, Phoenix-Scottsdale, AZ, 2002.

[11] C. M. Pun and X. C. Yuan, "Robust segments detector for de-synchronization resilient audio watermarking," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 11, pp. 2412-2424, 2013.
[12] D. Megias, J. Serra-Ruiz, and M. Fallahpour, "Efficient self-synchronised blind audio watermarking system based on time domain and FFT amplitude modification," *Signal Processing*, vol. 90, no. 12, pp. 3078-3092, 2010.
[13] B. Chen and G. W. Wornell, "Quantization index modulation: a class of provably good methods for digital watermarking and information embedding," *IEEE Transactions on Information Theory*, vol. 47, no. 4, pp. 1423-1443, 2001.
[14] W. N. Lie and L. C. Chang, "Robust and high-quality time-domain audio watermarking based on low-frequency amplitude modification," *IEEE Transactions on Multimedia*, vol. 8, no. 1, pp. 46-59, 2006.
[15] M. Fallahpour and D. Megias, "Audio watermarking based on Fibonacci numbers," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 8, pp. 1273-1282, 2015.
[16] J. Q. Zhang, H. X. Wang, and X. H. Li, "Robust audio watermarking algorithm based on neighborhood averaging method," *Journal of the China Railway Society*, vol. 34, no. 7, pp. 43-48, 2012.
[17] B. Lei, Y. Soon, F. Zhou, Z. Li, and H. Lei, "A robust audio watermarking scheme based on lifting wavelet transform and singular value decomposition," *Signal Processing*, vol. 92, no. 9, pp. 1985-2001, 2012.
[18] B. Lei and Y. Soon, "A multipurpose audio watermarking algorithm with synchronization and encryption," *Journal of Zhejiang University SCIENCE C*, vol. 13, no. 1, pp. 11-19, 2012.

Jinquan Zhang https://orcid.org/0000-0001-6889-2568

He received his Ph.D. from Southwest Jiaotong University, Chengdu, China, in 2013. His research areas include audio watermarking, cryptography, digital signature and network security. He is an Assistant Professor in School of Cybersecurity, Chengdu University of Information Technology.

Bin Han https://orcid.org/0000-0002-6201-944X

He received the B.S. degree from the Northeast Normal University of China, Changchun, in 1997 and the M.S. degree from the University of Electronic Science and technology, Chengdu, in 2006, both in computer science. He is an Assistant Professor in School of Cybersecurity, Chengdu University of Information Technology. His research interests include information hiding, digital watermarking and network engineering.