A Digital Watermark for Stereo Audio Signals Using Variable Interchannel Delay in High Frequency Bands

Kazuhiro Kondo and Kiyoshi Nakagawa
School of Science and Engineering, Yamagata University
4-3-16 Jonan, Yonezawa, Yamgata 992-8510, Japan
{kkondo,nakagawa}@yz.yamagata-u.ac.jp

Abstract

We propose a watermarking algorithm for stereo audio signals which embeds data using delay values of the high frequency channel signals. Since the stereo image perception of the human auditory system is known to be relatively insensitive to phase in high frequency regions, we replace the high frequency components with one middle channel, and embed data as delay between the channels. Blind detection of embedded data is possible using correlation between high frequency channels at the detector. Embedded data rate was 7 to 30 bps depending on the host audio signal. Embedded data was detected with little or no errors for added noise at 20 dB SNR and above. MP3 and AAC coders were shown not to affect the embedded data as well. The embedded audio quality was shown to be sound dependent, with quality equivalent to MP3 coded audio with some audio, but significantly lower for other sources.

1 Introduction

Although the volume of research in information hiding (watermarking) has been in images and video [1], we recently see new proposals for information hiding in speech and audio [2]. Most of these take advantage of the human auditory system (HAS) in order to hide information into the host speech or audio signal without causing significant perceptual disturbances.

Phase coding has been regarded as an effective method to embed data with minimum impact on the perceived quality [1]. Phase of the original audio is replaced with reference phase according to the data. Since the HAS is known to be relatively insensitive to small phase distortions, this method can potentially embed data without affecting the host signal quality significantly. In this paper, we use this property to code the embedded data in the relative phase between two stereo channels.

In the next section, the proposed watermark algorithm is described. In section 3, the watermark evaluation procedures as well as its results are described. Finally, in section 4, some discussions and conclusions as well as suggestions for future research are given.

2 Digital watermarking of stereo audio signals

In this section, we propose an audio watermark which embeds data by delaying either the left or the right channel of the stereo high frequency signals. It is well known that spatial localization of sound sources in HAS depends on both amplitude and phase characteristics for low frequency, but mostly on amplitude characteristics for high frequency [3]. This property of HAS is exploited in the intensity stereo coding used by some of the audio codecs, *e.g.* MPEG 1 Layer 3 (MP3) coders. We will also take advantage of this property, and alter the phase of the high frequency band in the stereo channels according to data. The amplitude envelope of these signals will be preserved in order to keep this alteration unperceivable.

2.1 Embedding

Fig. 1 shows the proposed watermark embedding algorithm. In this algorithm, the high frequency channels are mutually delayed by a fixed number of samples according to the embedded data. Input signal is first split into two subbands using the subband split filter. The cut-off frequency was set to approximately one tenth of the full bandwidth, *i.e.* approximately 2 kHz. We used a 128 tap FIR filter designed using the FFT windowing method. The left and right channel high band signal power, P_{LH} and P_{RH}, is calculated on a frame-by-frame basis. Then, the high band signals are averaged to give one middle high frequency channel. This signal will be copied for both left and right channels, but will be delayed on a frame-by-frame basis according to the

978-0-7695-3278-3/08 $25.00 © 2008 IEEE
DOI 10.1109/IIH-MSP.2008.35

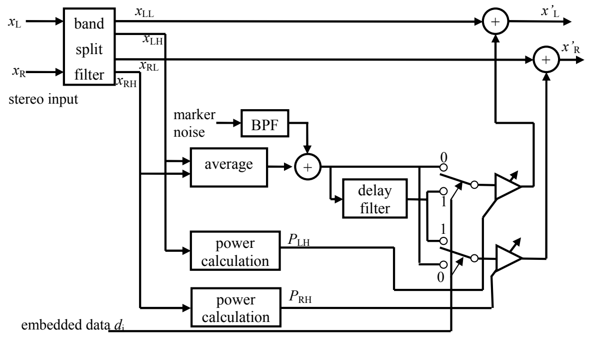


Figure 1. Proposed watermark embedding scheme.

embedded data. We chose to delay the left channel to mark a '0', and delay the right channel to mark a '1'. A frame was divided into two equal parts, and only the latter half of the frame was delayed according to the data. This is shown in Fig. 2. Thus, the first half of the frame is never delayed. We chose this arrangement to enable frame synchronization. If we embed a known data pattern regularly, *e.g.* consecutive zeros, there will always be a transition in the left channel delay at the middle and end of a frame, which should easily be detectable using correlation functions, for instance.

The amplitudes of each channel were then adjusted to match the power with the original signals, P_{LH} and P_{RH}. These signals are then added back to the low frequency components to obtain full bandwidth stereo audio signals.

We found that some of the frames did not contain enough high frequency components. Obviously, we are not able to encode data into these frames. We simply chose not to embed data in these frames. No delay was introduced in these frames. An example is shown in the third frame in Fig. 2.

The added delay should be small enough to be unperceivable. We empirically chose a maximum delay D of 3 samples. Even with this small amount of delay, abrupt changes are noticeable. Therefore, we smoothed the delay changes introduced according to a raised cosine-like pulse pattern. Non-integer delays were introduced using a sinc delay filter with impulse response $h(n)$:

$$h(n) = \frac{sin(\pi * (n - 1 - \tau))}{\pi * (n - 1 - \tau)} \quad (1)$$

where n is the sample number, τ is the added delay in samples, which can be fractional.

Also, as will be described in the next section, we will use correlation to detect delay between channel signals. In other words, we compare the cross correlation between channels with delay and without delay. If the former is larger than the latter, we detect embedded data. However, in some frames, the host signal itself showed relatively flat correlation, even for long lags. In these frames, we will not be able to detect delays using correlation. Thus, we added a small amount of low frequency noise as markers to enhance the difference between correlation with no delay and with delay. The amount of the added noise was empirically chosen as 10 % of the original frame power. However, even with the addition of markers, some frames do not display enough correlation difference. We chose not to embed data in these rare frames altogether.

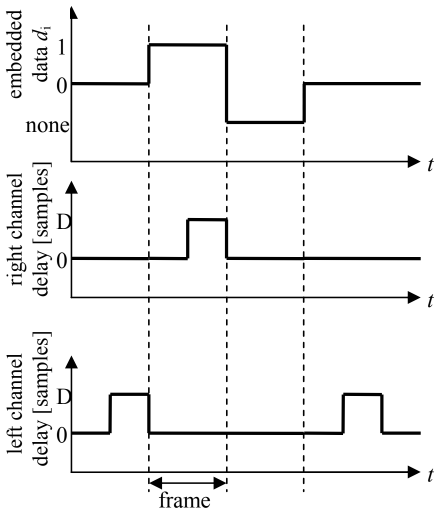


Figure 2. Example of embedded data and channel delay.

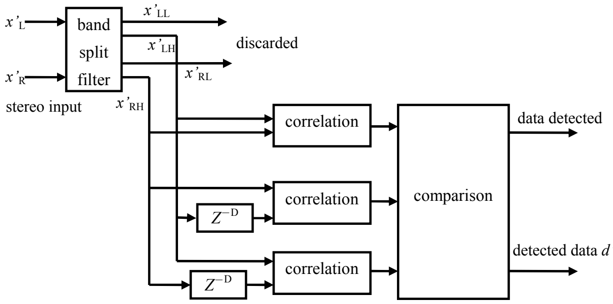


Figure 3. Proposed watermark detecting scheme.

2.2 Detection

Fig. 3 shows the proposed watermark detection algorithm. This algorithm does not require the original source as reference, and thus is blind detection. The input stereo audio signal with embedded data is first split into subbands using the same band split filter as in the embedding process. The low frequency portions are discarded. The high

Table 1. Embedded bits and bit rate.

Source	Length [samples]	Embedded bits	Bit rate [bps]
abba	667500	265	17.5
beethoven	275500	184	29.4
mozart	627500	101	7.1
piano	544000	90	7.3
trumpet	611500	132	9.5

frequency components are delayed by D samples. Frame boundaries are synchronized. Three normalized correlation values for the latter half of the frame are calculated: (1) left and right channel correlation with no delay, (2) correlation between delayed left channel and right channel with no delay, and (3) correlation between delayed right channel and left channel with no delay.

If the first correlation is the largest, we assume no data has been embedded. If the second correlation is the largest, we detect that a ’0’ has been embedded, and if the third correlation is the largest, we detect a ’1’. If the high energy components are below the threshold, we assume that no data has been embedded. Thus, the detection process is a relatively simple procedure. The frame synchronization may be somewhat expensive, but this is not required once synchronization has been established. The frame synchronization can be established by calculating a sample-by-sample correlation analysis near the embedded sync pattern.

3 Experimental evaluation

We evaluated the proposed algorithm using four short stereo audio samples; one rock instrumental (**abba**), and three classical pieces (**beethoven**, **mozart**, **piano**, and **trumpet**). All sources were CD quality, sampled at 44.1 kHz, stereo, 16 bits/sample. The embedding frame rate was set to 1000 samples, or about 23 msec.

3.1 Embedded data bit rate

Embedded number of bits for each source is shown in Table 1. The bit rate is apparently source dependent. Audio with high energy seems to be able to embed more data. This probably is because we do not embed data in frames with low high-frequency energy.

3.2 Robustness Evaluation

We also evaluated the robustness of our algorithm to two types of disturbances; additive white Gaussian noise and audio codecs. We embedded three bit patterns in our host signals: **pattern 1** was random bit pattern with same average number of ’0’ and ’1’, **pattern 2** was alternating ’0’ and ’1’, and **pattern 3** was alternating ’1’ and ’0’, *i.e.* phase-inverted **pattern 2**.

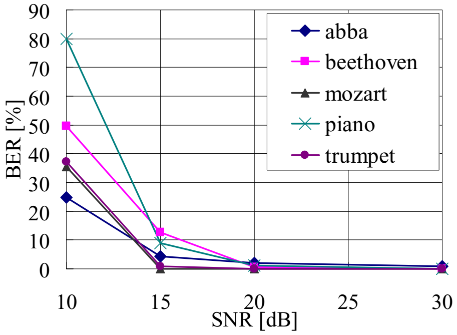


Figure 4. BER for random embedded bit pattern (pattern 1) with additive noise.

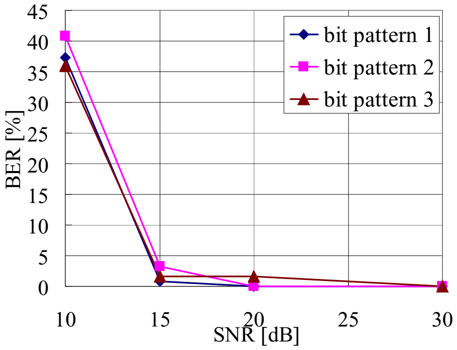


Figure 5. BER for various embedded bit pattern with additive noise (trumpet).

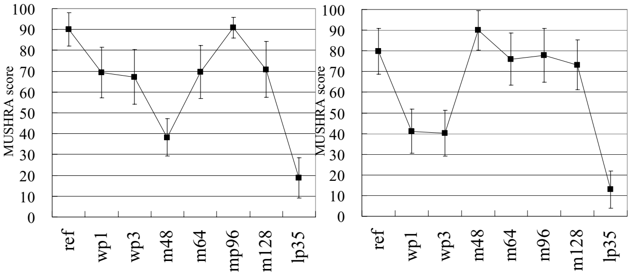
Since the proposed algorithm can show three error modes (substitution, where ’0’ is replaced with ’1’ and vice versa, deletion, and insertion errors), we evaluated bit errors after aligning the detected bit pattern using dynamic programming. We used sclite from NIST’s scoring package, freely available on the net. However, most of the errors turned out to be deletion errors. Only in rare cases, substitution and insertion errors were seen.

Fig. 4 shows the signal to added noise ratio and the bit error rate of all sources when **pattern 1** was embedded. There are differences in BER below SNR of 15 dB, but all sources show BER close to 0 above 20 dB SNR. Note that added noise larger than that giving 20 dB SNR is quite audible and renders the audio data worthless. Fig. 5 compares the BER for **patterns 1** through **3**. No significant difference is seen with the embedded data pattern.

We also tested the robustness of the embedded data when coded with MPEG audio codecs. We encoded each source with MPEG 1 Layer 3 (MP3) coding and MPEG 4 AAC codecs. Lame version 3.98 beta 5 with fixed rate of 128

Table 2. BER of MP3 and AAC encoded audio.

Source	BER [%]	
	MP3	AAC
abba	0.4	0.0
beethoven	0.0	0.0
mozart	1.0	0.0
piano	1.1	0.0
trumpet	0.0	0.0



(a) abba (b) piano

Figure 6. Subjective quality test results

kbps and joint stereo coding was used to code sources to MP3 and back to PCM. Apple Computer's iTunes version 7.4.3.1 was used to code sources to AAC and back. The AAC codec uses the Low Complexity Profile AAC to code sources to a fixed rate of 128 kbps. Table 2 shows the BER when random bits were embedded. In the majority of cases, no bit errors were seen, suggesting that the algorithm is robust to MPEG coders.

3.3 Subjective quality

The embedding process introduced a small noticeable quality alteration for some of the sources, while for others, no degradation was perceived. We conducted the MUSHRA subjective tests [4] to quantify the degradations. We tested audio clips stated above with the proposed data embedding method with bit pattern 1 (**wp1**) and bit pattern 3 (**wp3**). For reference, we included the original audio (**ref**), 3.5 kHz low pass filtered audio as anchor (**lp35**), and MP3 transcoded audio at 128 kbps(**m128**), 96 kbps (**m96**), 64 kbps (**m64**), and 48 kbps (**m48**), respectively. Twelve subjects participated in the tests.

Fig. 6 shows the MUSHRA scores and the 95 % confidence interval for **abba** and **piano**. The embedded audio showed approximately the same quality as MP3 at 64 and 128 kbps for **abba**. For **piano**, however, embedded audio showed significantly lower quality than MP3 at any rate. The quality for other sources fall between these two. The embedded quality does not seem to differ by embedded bit patterns. However, the quality is apparently not transparent, and needs improvement for some sources.

4 Conclusion

We proposed an audio watermarking algorithm that exploits the fact that the perception of the auditory system for stereo audio images is relatively immune to phase in the high frequency regions. The high frequency stereo signals were replaced with a single average middle channel. Data is encoded into the delays between the channels. The detection of embedded data can be achieved without the original signal, *i.e.*, blind detection. Correlation between the right and left channel high frequency signals are compared to detect the signals. The proposed algorithm showed embedded data bit rate between 7 and 30 bps depending on the host audio signal. The algorithm was also tested for robustness to added noise and MPEG coding. Little or no errors were seen for additive noise with SNR above 20 dB. MPEG 1 layer 3 and MPEG AAC coding also do not seem to cause many errors. The subjective quality of the data-embedded audio was shown to be source dependent, with some sources showing equivalent quality with MP3 coded audio, while for others the quality was significantly lower.

The current proposal uses high frequency energy threshold to decide whether to embed data or not. This implies variable bit rate, and also gives rise to insertion and deletion errors. Thus, we would like to explore algorithms which allow us to embed in every frame, enabling us to embed at a fixed rate. The proposed algorithm also uses blind detection. However, if the reference host signal is available, direct correlation can be used with the reference high frequency signals for improved robustness in detection. There is no need to change the embedding process. However, we can possibly increase the embedded data bit rate if we assume informed (non-blind) detection by changing the embedding process, perhaps by increasing the frame rate, or using multiple delay values.

References

[1] W. Bender, D. Gruhl, N. Morimoto, and A. Lu. Techniques for data hiding. *IBM System Journal*, 35(3 & 4):313–336, 1996.

[2] N. Cvejic and T. Seppanen, editors. *Digital Audio Watermarking Techniques and Technologies*, chapter 1. Information Science Reference, Hershey, PA, 2007.

[3] J. D. Gibson, T. Berger, T. Lookabagh, D. Lindberg, and R. L. Baker. *Digital Compression for Multimedia*, chapter 11, page 402. Morgan Kaufmann, San Francisco, CA, 1998.

[4] ITU recommendation BS.1534-1: Method for the subjective assessment of intermediate quality level of coding systems, 2003.