Method to Improve Watermark Reliability

Adam Brickman

EE381K - Multidimensional Signal Processing

May 08, 2003

ABSTRACT

This paper presents a methodology for increasing audio watermark robustness. The method exploits the multidimensional nature of a stereo audio file, which provides two channels to hide data. By incorporating redundant information along both channels, the total number of errors due to attacks should be less than or equal to that produced by any one method alone.

1. INTRODUCTION

Audio watermarking involves embedding data as additional information into an audio file. The most common application is for copyright protection to resolve piracy disputes, although some have doubts about the feasibility of this applications [1]. A watermark should not be detectable by statistical means [2]. In addition, knowing the watermarking scheme should not help a user extract the hidden data [3]. Any process that may damage a watermark is called an "attack." The goal of most watermarking schemes is to be resistant to as many types of attacks as possible and to "reliably" hide information. That is, it should be able to consistently extract the hidden message. This goal of this paper is to present a technique that helps minimize the number of errors caused by various attacks.

2. REVIEW OF PREVIOUS WORK

A large class of audio watermarks fall into the "spread spectrum" category. The audio to be watermarked is modeled as a random vector whose elements x_i are independent identically distributed

Gaussian random variables [4]. The watermark is a pseudo-noise sequence of "chips." Each chip w_i has value ±1. The marked signal y can be expressed as a weighted combination of these chips:

$$y = x + \delta w \quad (1)$$

A watermark w is detected by correlating a received signal z with w :

$$C(z, w) = E[z \cdot w] + \mathrm{N}(0, \tfrac{\sigma_x}{\sqrt{N}}) \quad (2)$$

A watermark is considered present if the correlation is above a specified threshold.

Cvejic *et al.* propose a spread spectrum method that incorporates the human auditory system's temporal sensitivity [5]. Boney *et al.* [6] also use temporal masking, and incorporate the MPEG psychoacoustical frequency masking model [7]. The authors claim that their technique embeds the maximum amount of information while remaining perceptually inaudible. Bassia *et al.* [8] use a similar embedding approach. However, watermark inaudibility is achieved via noise shaping using a Hamming window. Kirovski *et al.* [9] increase robustness to detector desynchronization attacks (a major problem with spread spectrum coding) via temporal beat detection in the host audio file. Although repeated chip coding can help alleviate synchronization problems [10], it facilitates watermark estimation attacks. Audio watermarking algorithms have also been accomplished in the frequency domain. Kuo *et al.* [11] present a form of covert audio watermarking using phase modulation for proof of ownership applications.

Watermarking can be related to a communications channel problem. The optimal attack strategy is the solution of a particular rate-distortion problem, and the optimal hiding strategy is the solution to a channel coding problem [12]. The channel capacity is defined as the maximum mutual information between an input X (the watermarked data) and output Y (3):

$$C_{chan} = \max_{p(x)} I(X:Y) = \max[h(Y) - h(Y|X)] \qquad (3)$$

The maximum is taken over all possible distributions p(x), and the term h(X|Y) represents information loss due to channel noise, which is essentially due to the combination of the original audio and signal processing procedures.

3. Description of Process

Although there are many individual watermarking methods, very little (if any) attention has been given to the use of more than one in an overall watermarking scheme. Given a stereo audio file, we may individually watermark the left and right audio channels. It is best to use two methods that have non-overlapping robustness to various attacks such that where one method may be weak the other is strong. In this manner, the combined result should be more robust than either of the individual parts. A flowchart of the overall procedure is illustrated in Figure 1.

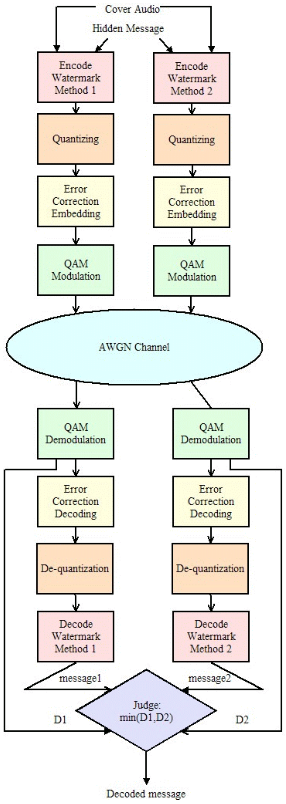
Cover Audio
Hidden Message
Encode Watermark Method 1
Encode Watermark Method 2
Quantizing
Quantizing
Error Correction Embedding
Error Correction Embedding
QAM Modulation
QAM Modulation
AWGN Channel
QAM Demodulation
QAM Demodulation
Error Correction Decoding
Error Correction Decoding
De-quantization
De-quantization
Decode Watermark Method 1
Decode Watermark Method 2
message1
message2
D1
Judge: min(D1,D2)
D2
Decoded message

Figure 1: Flowchart

3.1 Pre-Channel Processing

Pre-channel processing includes all the steps before QAM (quadrature amplitude modulation) (see Fig. 1). Watermarking schemes are discussed in section 4. Two different methods of error correction were used on separate trials: a (7,4) Hamming code and a integer input (15,4) Reed-Solomon code. Quantization correspondingly took one of two forms: (1) bit quantization or (2) integer amplitude level quantization.

3.2 Channel Processing

The data was sent through the channel via 16-ary QAM, chosen for its simple rectangular constellation lattice. QAM modulation employs two quadrature (90° out of phase) carriers [13]. The transmitted waveforms have the form

$$u_m(t) = A_{mc} g_T(t) \cos 2\pi f_c t + A_{ms} g_T(t) \sin 2\pi f_c t, m = 1,2,\dots, M \quad (4)$$

where $\{A_{mc}\}$ and $\{A_{ms}\}$ are sets of amplitude levels obtained by mapping *k*-bit sequences into signal amplitudes. Thus, the transmitted waveform has an in-phase (cosine) and a quadrature (sine) part. The received signal will have a similar form to the transmitted signal with the addition of noise and possibly a carrier phase change. We can determine its corresponding constellation point by performing a "nearest neighbor" calculation. That is, given a received vector r we select the signal point s_m corresponding to the smallest value of the computed Euclidean distance metric:

$$D(r, s_m) = \left| r - s_m \right|^2 \quad (5)$$

3.3 Post-Channel Processing

After the data has been modulated, passed through the channel and demodulated, we must "undo" the steps taken in the pre-channel processing. We decode the error correction to yield a sequence of received bits (Hamming) or received integers (Reed-Solomon). We return this data into floating point format and each watermarking scheme decodes the hidden message. Because of the channel effects channel, this message may have been damaged or destroyed. If the watermarking algorithms are fed the same input data and message to hide, then they should produce identical outputs.

3.4 Determining the hidden message

If the determined message values of each watermarking scheme match, then there is no dispute and either message value may be passed along to the final output. The arbitration of conflicting message values is handled by a "judge" routine. It makes its decision based on how close the received QAM signal vectors were to their nearest constellation points (see (5)). The method with the lower average Euclidean distance wins the dispute and has its message value appended to the final output message. When all message bits have been compared, the process is complete.

4. WATERMARKING METHODS EMPLOYED

Although many image watermarking implementations are readily available for download on the Internet, remarkably few are available for audio. Of these, only two programs executed without generating an error. *Steghide* [14] was able to successfully embed a hidden message. However, if any attack was introduced, decoding produced an error and aborted before completion. A demo version of *Invisible Secrets*, a professional watermarking program, appeared very robust to many attacks. However, if attacks were too severe, it also aborted. It would be impossible to use these programs and obtain detailed information about the number of bit errors. Instead, two methods of watermarking were implemented using MATLAB software, detailed below.

4.1 Echo Hiding

One method developed by Gruhl *et al.* [15] proposed to encode bits by introducing a small, imperceptible echo to the file. We convolve an echo kernel with the original signal to produce an echo. The echo kernel consists of two impulses separated by a time offset. The offset amount determines whether we embed a “1" or a “0.” Through trial and error, echoes of 1.3ms and 1.0ms yielded good decoding results with minimal audibility. The amplitude of the echo kernel may also be adjusted. A higher amplitude means a stronger echo. When the echo kernel amplitude was less than .5, very few listeners could hear any difference between the original and echoed signal. Stronger amplitudes produced a more resonant, “richer” sound. The cover audio is segmented and a bit is embedded into each segment. To help decrease audibility, a mixing window cross-fades between adjacent segments. Decoding essentially involves examining the magnitude of the autocepstrum of the stego (echo-embedded) signal at the locations of the kernel delays. The autocepstrum is calculated by (6)

$$F^{-1}\{\ (\ln F(x[n]))^2\ \} \quad (6)$$

where F and F^{-1} denotes the Fourier transform and inverse Fourier transform respectively. This method transforms convolution into a linear operation, lowering the computational complexity to O(n*log(n)).

4.2 Least Significant Bit Hiding

Another method places message bits into cover audio by modifying the least significant bits of the audio. We developed a scheme that places bits into the *m*th bit of the cover audio, where *m* is a parameter that ranges from 1 (MSB) to 16 (LSB). This method has extremely low computational complexity, on the order of O(n). To allow as fair a comparison between watermarking methods as possible, the cover audio was segmented as it was in echo hiding and bits were placed at the first location of each segment. Thus, the embedded bit locations were known in advance. This violates the provision that watermarks should be statistically invisible; we can technically consider this method more of a "data-hiding" than a watermarking algorithm. Decoding simply involves taking values at these known locations and extracting the desired bit. Interestingly, bits encoded down to the 10th bit location could not be heard by human observers.

5. System Implementation

The system described in section 3 was achieved using a combination of MATLAB and Simulink software. Audio data first passes through watermarking and quantization code. Next, the "channel processing" operations are performed in a sequence of three steps, detailed below.

5.1 Channel Coding

As shown in Fig. 2, the watermarked, quantized audio is buffered to produce length-4 words that were either sent through a Hamming encoder (shown) or a Reed-Solomon encoder.

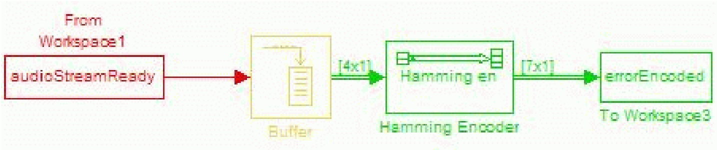

Figure 2 - Channel Coding

Care must be taken to ensure that the encoders are given the appropriate data format.

5.2 Channel Transmission

The error correction coded data is passed to the channel transmission block. The 16-QAM modulator and demodulator requires the inputs to be between 0 and 15. Thus, either 4 bits from the Hamming code or one integer in the above range from the Reed-Solomon code are taken as inputs to the modulator. The channel is modeled as additive white Gaussian noise with an adjustable signal to noise level. Note the "encodedQAMNoisy" output of Figure 3; this is used in the judging step to arbitrate watermark disputes, as described in section 3.4.

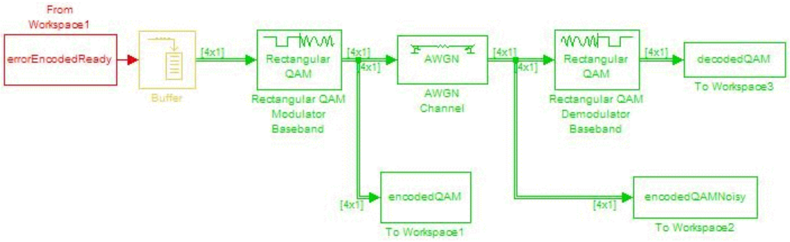

Figure 3 - Channel Transmission

5.3 Post-Channel Processing

The error decoding process resembles Figure 3 except that we substitute a decoder in place of the encoder. To decode the watermark, we must be careful to consider the effects of buffering in each stage, which may cause delays in the output. This was incorporated into decoding algorithms that transformed the received, error corrected channel data into a format suitable for watermark extraction. The final judging stage compared the decoded messages from each watermark algorithm. Only one bit per segment is important to LSB decoding whereas all bits in a given segment are needed for echo decoding. Thus, for echo hiding, the mean distance of *all* the QAM mapped points is calculated, whereas only one distance value is calculated for LSB hiding. For each segment that has a conflicting watermark message, the method with the lower distance value passes its message bit to the final output.

6. RESULTS

Tests were conducted on a 5 second CD-quality sample of classical music. A total of twenty bits of an alternating one-zero pattern were embedded into this cover audio. This is the most difficult situation for the echo hiding method. To compensate, the echo hiding parameters were set to provide the highest possible success rate. Hamming error correction code performed slightly better than the Reed-Solomon code on average. Attacks were either simulated by MATLAB in the AWGN channel or produced with Sonic Foundry's Sound Forge software. Table 1 shows representative results from a variety of attacks.

In general, the echo hiding method was far more robust to attacks. It did, however, produce one bit error even when there was no channel attack present. It also (surprisingly) suffered more from the additive white Gaussian channel noise. The LSB-10 method with Hamming coding (hiding into the 10th bit) was extremely robust to this form of attack. Clearly, the channel made it difficult to discern echoes when the SNR dropped to 10dB or below. On average, the Hamming error correction code performed slightly better than the Reed-Solomon code. We can see that embedding into the 16th bit (LSB hiding) suffered greatly to all audio attacks. A good alternative is to instead embed into the 10th bit, which is still inaudible yet exhibited far more robustness. The "judging" technique achieved its goal by reducing, or at worst not increasing, the total number of message errors.

7. Future Work

The proposed method of increasing watermark reliability was successful in helping minimize the number of bit errors. However, the procedure is extremely computationally intensive and requires large amount of memory storage. This problem was manageable for 5 seconds of data; it is not feasible for an entire CD. In the future, a faster, more efficient algorithm should be developed. This method exploited the dimensionality of a stereo audio file. It may naturally be extended to higher dimensional files such as Dolby Surround 5.1 to possibly produce even better results.

Table 1 - Total Number of Errors Produced by Various Attacks

Channel/Attack Properties	EchoHiding	LSB - 10	LSB - 16	Post-judged Echo vs. LSB10	Post-judged Echo vs. LSB16
AWGN (SNR = 20dB)	1	0	0	0	0
AWGN (SNR = 15dB)	2	0	0	0	0
AWGN (SNR = 10dB)	10	0	0	7	7
AWGN (SNR = 1dB)	7	0	3	5	4
Normalization	1	5	14	N/A	N/A
6 dB BassBoost	4	6	10	N/A	N/A
Resampled - 44090 Hz.	1	0	0	N/A	N/A
16 ms Reverb (-30dB)	1	0	0	N/A	N/A
DC Offset (+1)	1	0	20	N/A	N/A

REFERENCES

[1] C. Herley, “Why Watermarking Is Nonsense,” *IEEE Signal Processing Magazine*, pp. 10-11, Sep. 2002.

[2] S. Voloshynovkiy, S. Pereira, T. Pun, J.J. Eggers, J.K. Su, “Attacks on Digital Watermarks: Classification, Estimation-Based Attacks, and Benchmarks,” *IEEE Communications Magazine*, vol. 39 no. 8, pp. 118-126, Aug. 2001.

[3] F.A.P. Petitcolas, M. Steinebach, F. Raynal, J. Dittmann, C. Fontaine, S. Seibel, N. Fates, L.C. Ferri, “StirMark Benchmark: Audio Watermarking Attacks,” *Proc. Information Technology: Coding and Computing*, pp. 49-59, Apr. 2001.

[4] D. Kirovski and H.S. Malvar, “Spread Spectrum Watermarking of Audio Signals,” *IEEE Transactions on Signal Processing*, vol. 51, no. 4, April 2003.

[5] N. Cvejic, A. Keskinarkaus, T. Seppanen, “Audio Watermarking Using m-Sequences and Temporal Masking,” *Proc. IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*, pp. 227-230, 2001.

[6] L. Boney, A.H. Tewfik, K.N. Hamdy, “Digital Watermarks for Audio Signals,” *Proc. IEEE International Conference on Multimedia Computing and Systems*, pp. 473-480, June 1996.

[7] P. Noll, “MPEG Digital Audio Coding,” *IEEE Signal Processing Magazine*, vol. 14, no. 5, pp. 59-81, Sep. 1997.

[8] P. Bassia, I. Pitas, N. Nikolaidis, “Robust Audio Watermarking in the Time Domain,” *IEEE Transactions on Multimedia*, vol. 32, no. 2, pp. 232-241, June 2001.

[9] H. Attias and D. Kirovski, “Audio Watermark Robustness to Desynchronization via Beat Detection,” *Proc. Information Hiding Workshop,* 2002.

[10] D. Kirovski and H. Malvar, “Robust Spread Spectrum Audio Watermarking,” *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, pp. 1345-8, 2001.

[11] S. Kuo, J.D. Johnston, W. Turin, S.R. Quackenbush, “Covert Audio Watermarking Using Perceptually tuned Signal Independent Multiband Phase Modulation,” *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing,* vol. 2, pp. 1753-1756, May 2002.

[12] P. Moulin and J.A. O’Sullivan, “Information-Theoretic Analysis of Watermarking,” *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 6, pp. 3630-3633, 2000.

[13] J.G. Proakis and M. Salehi, Communications Systems Engineering. Prentice Hall, Inc., NJ, 1994.

[14] S. Hetzl, “The Steghide Website,” http://steghide.sourceforge.net/.

[15] D. Gruhl, W. Bender, A. Lu, “Echo Hiding,” *Proc. Info Hiding*, pp. 295-315, 1996.