ON COMBINING WATERMARKING WITH PERCEPTUAL CODING

Jack Lacy, Schuyler R. Quackenbush, Amy R. Reibman, David Shur, and James H. Snyder

AT&T Labs
Florham Park, NJ; Red Bank, NJ; Holmdel, NJ
{lacy,srq,amy,jhs}@research.att.com, shur@att.com

ABSTRACT

A watermark is a data stream inserted into multimedia content. It contains information relevant to the ownership or authorized use of the content. A watermark which could be recovered without *a priori* knowledge of the identity of the content could be used by web search mechanisms to flag unauthorized distribution of the content. Since media will be compressed on these sites, a mark detection algorithm that operated in the compressed domain would be useful. We describe in this paper a watermark algorithm which operates in the compressed domain and does not require a reference.

1. INTRODUCTION

Electronic distribution of multimedia content is an important byproduct of the confluence of recent technological advances. Compression algorithms that preserve audio and video quality while reducing bit rate dramatically, increasing network bandwidth, higher density storage devices, and network search engines, all taken together support network services which can revolutionize the distribution of music and video.

However, content owners naturally wish to maintain control over their wares. To protect their intellectual property (IP), an integrated system design is necessary [3]. The basic system consists of three major building blocks. First, compressed content is stored in a cryptographic container before distribution to users. Second, a flexible licensing mechanism answers questions about the trustworthiness of those seeking access to the content. Third, watermarks are embedded in the content in an imperceptible fashion so that the content can be identified if the cryptographic container is breached. Finally, a secure system design integrates these three components.

In section 2, we identify different uses of watermarks and describe how these uses affect algorithm choice. We then focus in section 3 on a watermarking technique that can be detected in the compressed domain without a reference. The technique avoids a complete decode. Section 4 describes preliminary results obtained with this algorithm.

2. WATERMARKING

2.1 Watermarking Uses and Requirements

Watermarks typically serve one of three functions: identification of the origin of the content, tracing illegally distributed copies of the content, and disabling unauthorized access to the content. No one marking algorithm is best suited to all three functions, both because of complexity, and because different functions and different marking algorithms are resistant to different kinds of attacks. Indeed, we expect that any single piece of music or video will be marked with a variety of different algorithms.

For copyright identification every copy of the content can be marked identically, so the watermark can be inserted once prior to distribution. Ideally, detection should not require a reference because the search engine has no *a prioi* way to identify the work from which it must recover the mark. The watermark should be detectable inside an edited work in which the original content may be either shortened or abutted with other works. Not only must the watermark be short enough to be detected in a shortened version, but some means must be provided to synchronize the detection process so that the watermark can be located in the processed bitstream. Finally, the watermark must be robust to further processing. Any attempt to remove it, including re-encoding the content, should lead to perceptible distortion.

Transaction identification requires a distinct mark for each transaction. The primary challenge of point-of-sale marking is to move the content through the watermarking engine quickly; that is, the algorithm must be low complexity. One strategy is to insert the watermark in the compressed domain. Ideally mark insertion should increase the data rate very little. In contrast to copyright ownership, the watermark must be robust to collusion attacks.

We believe that disabling access to content is best performed by mechanisms other than watermarks. If a watermark is nonetheless used to disable access to content, the watermark recovery mechanism should be of low complexity. It should *not* be a protection of last resort, as disabling access clearly indicates, to anyone who can reverse-engineer the access mechanism, the location of the watermark.

2.2 Watermarking Models

Watermarks used in conjunction with compression algorithms fall into one of three classes: cleartext (PCM) marking, bitstream marking, and marking integrated with the compression algorithm. Each type has advantages and disadvantages, which we discuss briefly. More detail is given in a future paper [4].

Cleartext marking

Cleartext marking relies on perceptual methods to embed a data stream in a signal imperceptibly. The model for many cleartext marking algorithms is one in which a signal is injected into a noisy communication channel, where the audio/video signal is the interfering noise [2]. Because the channel is so noisy, and the

mark signal must be imperceptible, the maximum bit rates that are achieved for audio are generally less than 100bps.

A cleartext mark appears in all processed generations of the work, since by design the marking algorithm is both secure and robust in the face of typical processing. It is therefore well suited to identification of the work. There are two major disadvantages to cleartext marking. Because such algorithms compute a perceptual model, they tend to be too complex for point-of-sale applications. A potentially significant problem is that these algorithms are susceptible to advances in the perceptual models used in compression algorithms. Many cleartext marking algorithms have been reported, see e.g. [5].

Retrieval mechanisms for cleartext watermarks fall into two classes: reference necessary and reference unnecessary. In either case the mechanism for mark recovery is generally of high complexity. Further, if means for detecting these watermarks are embedded in a player, an attacker, by reverse engineering the player, may be able to identify and remove the marks. We feel that cleartext watermarks should *not* be used to gate access to content.

Bitstream marking

Bitstream marking algorithms manipulate the compressed digital bitstream without changing the semantics of the audio or video stream. For example, a data envelope in an MPEG-2 Advanced Audio Coding (AAC [1]) audio frame could contain a watermark, albeit one which could easily be removed. Bitstream marking is low-complexity so can be used to carry transaction information. However these marks cannot survive D/A conversion and are generally not very robust against attack; e.g. they are susceptible to collusion attacks. Because the mark signal is unrelated to the media signal, the bit rate these techniques can support can be as high as the channel rate. This type of mark can easily be extracted by clients and is thus appropriate for gating access to content.

Integrated marking

Integrating the marking algorithm with the compression algorithm avoids an 'arms race' between marking and compression. Since the perceptual model is available from the workings of the compression algorithm, integrated marking algorithms alter the semantics of the audio or video bitstream, thereby providing resistance to collusion attacks. Depending on the details of the marking algorithm, the mark may survive D/A conversion. An example of this approach is [6], which does not use perceptual techniques.

3. A WATERMARKING SYSTEM FOR MPEG AAC and MPEG VIDEO

3.1 Subthreshold marking

Our watermarking system, diagrammed in Figure 1, is a first generation system that combines bitstream and integrated water-marking. It can be configured to support the three marking functions mentioned above. It does not include but is compatible with use of a front-end cleartext marking algorithm as well. We assume that the cleartext original is not available to any parties except possibly auditors seeking to recover the watermark. In particular, the cleartext original is not available to attackers. The decompressed and marked content will generally be available to everyone.

In this paper, we describe only the integrated watermarking component. Figure 2 shows a simplified block diagram of a generic perceptual coder. Our marking technique involves the perceptual modeling and rate control, quantization, and noiseless coding blocks. In MPEG AAC spectral lines are grouped into 49 "scale factor" bands (SFB), each band containing between 4 and 32 lines. Associated with each band is a single scale factor, which sets the quantizer step-size, and a single Huffman table (AAC employs 11 non-trivial Huffman tables). The coefficient for each spectral line is represented by an integer (i.e. quantized) value. In MPEG video, a block consists of 64 coefficients, and each set (termed a macroblock) of 6 blocks has an associated quantization step-size Q_p. The same Huffman table is used for the coefficients for all Q_p values. As with audio, each coefficient is represented by an integer after quantization. Because the watermarking algorithms for audio and video are essentially identical, for consistency we use the audio terminology (scale factor) throughout when we are discussing techniques. When we discuss the results for video, we will use terminology specific to video.

Let $A = \{f_i, H_i, \{q_{ij}\}\}$ be the set of triples of scale factors f_i, Huffman tables H_i, and quantized coefficients $\{q_{ij}\}$. (Only one Huffman table is used in video.) We now describe three different methods for inserting a mark into the bitstream imperceptibly. We assume that we have selected some set of scale factor bands into which mark data will be inserted. We do not specify here a method by which SFB are chosen for marking; however for audio SFB encoded with the null Huffman table H_0 should probably not be marked. For video, zero coefficients should remain zero and not be modified. Hence the marking set will be dynamic. Let M be the set of indices associated with the set of SFB chosen for marking.

Method 1. Choose a set of multipliers $\{x_i=2^{N_i}: i\in M\}$. Modify the triple $\{f_i, H_i, \{q_{ij}\}: i\in M\}$ by dividing the scale factors by x_i, multiplying the quantized values $\{q_{ij}\}$ by $\{x_i\}$, and adding mark data $\{m_{ij}\}$ to the non-zero modified quantized values. Mathematically: $A\rightarrow A'$, where

$$\forall i: i\notin M,\ \{f_i', H_i', \{q_{ij}'\}\} = \{f_i, H_i, \{q_{ij}\}\},$$
$$\forall i: i\in M,\ \{f_i', H_i', \{q_{ij}'\}\} = \{f_i/x_i, H_i'', \{q_{ij}\times x_i + m_{ij}\}\},$$

where H_i'' is the smallest codebook that accommodates the largest value $q_{ij}\times x_i + m_{ij}$. Since the original scale factors were chosen perceptually, the resulting mark is imperceptible. A feedback mechanism similar to the one in [10] can be used to prevent modification of scale factors that would increase the bit rate significantly. Note that if the attacker can identify the frame and SFB containing the mark data, then that data can easily be removed. A possible attack on this method would be to run a perceptual model on the decompressed output. While it is unlikely that the perceptual model could indicate unambiguously every marked location, it seems likely that many could be identified.

Method 2. In this case, applicable only to audio, the mark data is indicated by two characteristics of the bitstream data. The indication that mark data is present is that the Huffman table

used to encode the SFB is not the table that would ordinarily be used. The value of the mark data bit (one bit per SFB) could be indicated in many ways; for example, if the SFB index is even the value is 0, otherwise 1. That is, $\{f_i, H_i, \{q_{ij}\}\} \rightarrow \{f_i, H_i', \{q_{ij}\}\}$.

There are two problems with this method. First, sectioning, a process by which codebooks are "promoted" to reduce bit rate, introduces similar changes in the choice of codebooks. That is, sectioning *itself* can erase the mark data indication. Second, this marking is particularly easy to identify, since an attacker looking at the bitstream can observe that the codebook used to encode the coefficients in the SFB is not the minimum codebook required. However by a sensible choice of SFB it is possible to insert mark data in a way that will not be modified by sectioning but which mimics the action of sectioning and so is (somewhat) less obvious to an attacker.

Method 3. The previous two methods were coupled to the encoder only via the overall bit rate limit. We now present a method for marking which is integrated with quantization. The mark is therefore difficult to remove without perceptible effects. As in Method 2, the fact that marking data is present is indicated by characteristics of the bitstream data. As in Method 1, we modify the scale factor f_i and the coefficients $\{q_{ij}\}$ by a factor x_i, but now all $\{x_i\}$ are close to unity. Let $\{v_{ij}\}$ be the set of spectral coefficients prior to quantization, and Q_i be the quantizer for SFB i, i.e. $\forall i \{q_{ij}\} = Q_i[\{v_{ij}\}]$. Then

$\{f_i, H_i, \{q_{ij}\}\} \rightarrow \{f_i', H_i', \{q_{ij}'\}\}$, where
$f_i' = f_i/x_i$
$q_{ij}' = Q_i[x_i \times v_{ij}]$
$H_i' = H_i$ or the next larger codebook
$x_i \cong 1$

Because the modification to the spectral coefficients occurs before quantization, the changes to the reconstructed coefficients will be below the perceptual threshold. If this change were introduced after quantization, the change in some quantized values would be greater than the perceptual noise floor. Equivalently, an attacker who modifies the quantized values to eradicate or modify the mark will be introducing energy changes that exceed the noise floor. Because the changes in step-sizes will be small, because not all coefficients will change, and because the attacker will not have access to the uncompressed cleartext source material, the attacker will generally not be able to identify those SFB which are used for marking. Further, the change in bit rate associated with marking should be small. In this method, the value of the watermark bit can be indicated in a variety of ways, e.g. it might take on the value of the LSB of the scale factor value, in which case a scale factor needs to be modified only if its LSB differs from the desired value. For both audio and video, the increase in bit count incurred by this method must be monitored.

3.2 Watermark Synchronization & Recovery

Generally watermark sequences are inserted a few bits per frame. The data to be carried by the stream is typically mapped into a marking sequence prior to embedding, where the characteristics of the mapping function depend on the type of attack expected. Indeed, since there may be a wide range of attacks, the data may be redundantly mapped in different ways in the hope that at least one mapping will survive all attacks. This leads to the issue of recognizing where a marking sequence begins. One approach is to use synchronizing codes. However the attacker may be able to identify these codes, and if the attacker can eliminate or damage the codes, recovery of mark data may not be possible.

In our system, synchronization is tied to frame boundaries. We modify the scale factors included at the beginning of the frame by modifying the LSBs so that they represent a sequence which contains one or more synchronization codes. Specifically, when we select a frame for synchronization insertion, and a scale factor LSB does not match (0 where a 1 is indicated, or a 1 instead of a 0), we decrement that scale factor and adjust all the coefficients in the SFB accordingly. Although the synchronization code can be damaged, random flipping of scale factor LSB by an attacker will introduce artifacts.

To recover the watermark we look for a synchronization code and recover the data appropriately to the method.

3.3 Audio results

To evaluate our audio watermarking algorithm we used AT&T's implementation of AAC. Watermark synchronization is indicated by the sequence comprising the LSB of the first 44 decoded scale factors in a long block. When the value of the LSB of a scale factor does not match the corresponding bit in the synchronization code then the scale factor is decremented and the spectral coefficients adjusted accordingly, resulting in perceptually irrelevant overcoding of the associated spectral data.

The following table shows the cost of carrying watermark data inserted by method 3 into *every frame* of an AAC bitstream for a stereo signal sampled at 44.1 kHz and coded at 96 kbps. Cost is expressed as increase in bits per frame (21.3 ms of audio) and increase in rate.

Table 1. Increase in audio bit-rate.

Method 3	increase in bits (per marked frame)	increase in rate
synchronization	5.2	0.25%
sync + 32 bits	9.0	0.44%

An important issue for any watermarking algorithm is the quality of the reconstructed signal following an attack which erases the watermark. We have simulated a naïve attack on this marking algorithm by zeroing all scale factor LSB, and find that this attack results in unacceptable distortion in the reconstructed audio signal.

3.4 Video results

Our baseline system for video compression uses a rudimentary perceptual model. A variance-based activity measure is used to select the quantization step-size for each macroblock as in step 3 of the MPEG-2 TM5 rate control [7]. We generate I frames every half second; all other frames are P frames. We inserted watermark data into both I and P frames, and present results taken from an average over two different 10 second sequences.

The first 44 macroblocks of a frame are used for synchronization as described in Section 3.2. The next several macroblocks (100 or 600 in the Table, of 1320) of a frame carry mark bits using

Method 3. For each macroblock, when the LSB of the step-size Q_p does not match, Q_p is decremented. However, a dead-zone is applied to the original Q_p to ensure that zero coefficients remain zero.

We have simulated a naïve attack on this algorithm by zeroing all scale factor LSB, and find that this attack results in a perceptible 1.6dB degradation in PSNR of the reconstructed video signal.

Table 2. Increase in video bit-rate.

Method 3	increase in bits (per marked frame)	increase in rate
synchronization	124	0.005%
sync + 100 bits	138	0.006%
sync + 600 bits	557	0.024%

4. CONCLUSIONS

This paper opens the discussion of the integration of watermarking with perceptual coding mechanisms. We have described a first generation technique which inserts data, typically a watermark, into an audio or video bitstream cooperatively with the compression algorithm. The data may be recovered with a simple decoding process. It is robust to attacks which modify bitstream scale factors, in the sense that damaging the mark produces perceptible artifacts.

5. REFERENCES

[1] IS 13818-7 (MPEG-2 Advanced Audio Coding, AAC), M. Bosi, K. Brandenburg, S. Quackenbush, M. Dietz, J. Johnston, J. Herre, H. Fuchs, Y. Oikawa, K. Akagiri, M. Coleman, M. Iwadare, C. Lueck, U. Gbur, B. Teichmann.

[2] J. Smith, B. Comisky, "Modulation and Information Hiding in Images", *Proc. First International Information Hiding Workshop*, LNCS 1174, Springer-Verlag, Cambridge, U.K., May/June, 1996, pp. 207-226.

[3] J. Lacy, D. P. Maher, and J. H. Snyder, "Music on the Internet and the Intellectual Property Protection Problem", *Proc. International Symposium on Industrial Electronics*, Guimaraes, Portugal, July 1997.

[4] J. Lacy et. al., "Watermarking and Intellectual Property Protection", submitted to the Second International Information Hiding Workshop, Portland, OR 1998.

[5] *Proceedings of the Fourth International Conference on Image Processing*, Santa Barbara CA, October 1997.

[6] F. Hartung and B. Girod, "Digital Watermarking of MPEG-2 Coded Video in the Bitstream Domain", *Proc. IEEE ICASSP*, pp. 2621-4, April 1997.

[7] MPEG video committee, "Test Model 5", ISO-IEC/JTC1/SC29/WG11 N0400, April 1993.

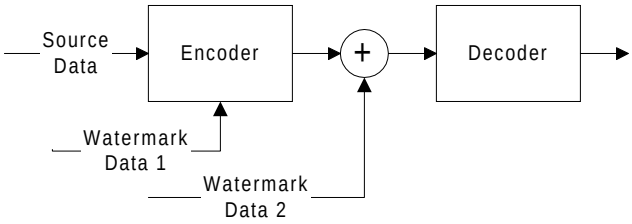

Figure 1. Watermarking System Block Diagram

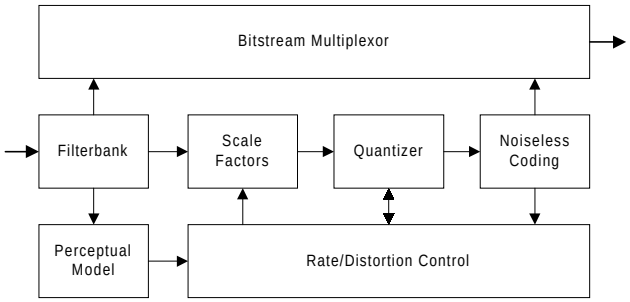

Figure 2. Generic Perceptual Encoder Block Diagram