H.264/AVC STREAM REPLACEMENT TECHNIQUE FOR VIDEO WATERMARKING

Dekun Zou and Jeffrey A. Bloom

Thomson Corporate Research
2 Independence Way, Princeton, NJ 08540, USA

ABSTRACT

This paper describes a method for applying a watermark directly to an entropy coded H.264/AVC stream. This method can be applied when the stream, or at least the I-frames, is entropy coded with CAVLC. Most of the hard work goes into building an embedding table during an analysis stage. This table identifies small segments of data in the encoded stream that will be replaced. The table also provides alternative values for each segment. The embedding process involves replacing each identified segment with one of the alternative values from the table. The choice of alternative is informed by the payload to be embedded. The resulting bitstream is guaranteed to be compliant with the H.264/AVC standard.

Index Terms— digital watermarking, data hiding, video watermarking, CAVLC, H.264/AVC

1. INTRODUCTION

There have been many papers published in recent years discussing methods for watermarking motion imagery in the pixel domain. A general summary of such *baseband* digital watermarking techniques can be found in the book Digital Watermarking [1]. For almost as long as researchers have examined baseband watermarking, they have also considered compressed domain watermarking techniques. These techniques are useful for applications in which the original unmarked works are already compressed and the watermarking process is time and resource constrained such that reconstruction, watermarking, and recompression (presumably using all of the motion estimates, quantization scale factors, and mode decisions from the original compressed file) would be too expensive.

Most published works describing a compressed-domain watermarking process do not, in fact, operate on the compressed bitstream. Instead, they apply entropy decoding and parsing of the bitstream to recover the syntax elements (coefficients, motion vectors, modes, etc.). The watermark is typically applied in this syntax domain and the result entropy coded. These methods are often called *partial decompression* methods (illustrated in Figure 1). Examples of approaches that target MPEG [2], MPEG-2 [3] and H.264/AVC [4] have been developed.

A new class of applications requires the watermarking to be applied directly to an entropy encoded bitstream without the entropy decode and re-encode steps. Figure 2 illustrates this direct stream embedding framework when offline analysis is possible. In this case, the bitstream will be modified directly where necessary to embed the watermark. This modification should only happen at limited locations. The challenge is to determine where and how the bitstream can be changed so that the modified bitstream is still a valid, compliant compressed bitstream, the resulting video is perceptually identical to the original video, and the modification (or information represented by the modification) can be recovered from the decoded, baseband pixel data.

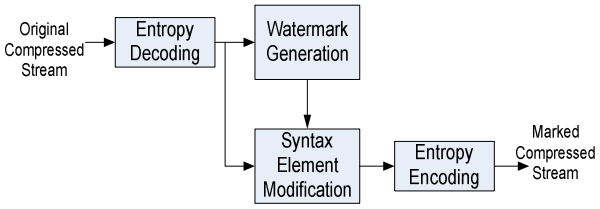

Figure 1. Syntax domain stream embedding

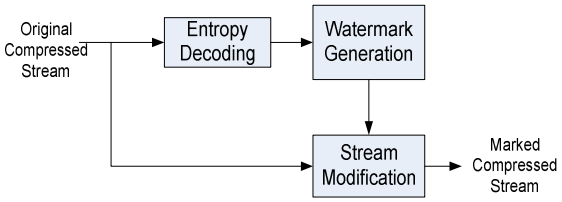

Figure 2. Direct stream watermarking

The work described in this paper has been motivated by an application of watermarking a motion image sequence represented by the H.264/AVC standard [5]. AVC is very competitive with other video compression methods in terms of compression efficiency. Both of the high definition DVD standards support AVC and it is expected to become more widely used in the future. This paper presents a byte replacement watermarking algorithm suitable for direct stream marking of H.264/AVC streams.

2. REAL-TIME BYTE REPLACEMENT WATERMARKING FRAMEWORK

The direct stream watermarking as illustrated in Figure 2 modifies the stream by directly modification as it passes through. Most bytes of the stream are passed unchanged, but some of the bytes are replaced with alternative values. The challenge becomes identifying the bytes to be replaced and generating the replacement values such that the resulting stream meets the 3 requirements of compliance, fidelity, and robustness. The computational burden lies in the entropy decoding and watermark generation steps which

might include a complex perceptual analysis to control the tradeoff between robustness and fidelity. In many applications, including those that motivated our work, the original compressed stream can be preprocessed off-line. Thus, we present the entropy decoding and watermark generation processes as an analysis step. The output of the analysis step is the metadata needed for stream modification watermarking. This framework is illustrated in Figure 3 with the dashed box representing the analysis step.

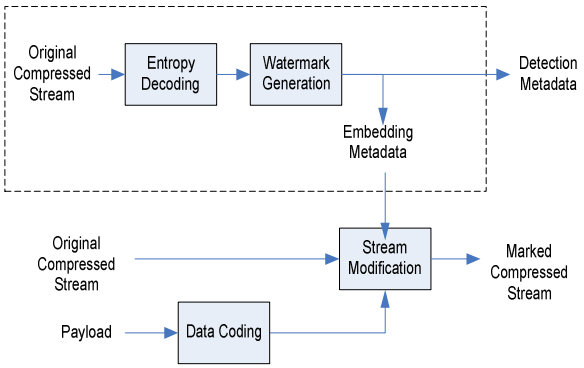

Figure 3. Real-time stream replacement watermarking

The analysis can be computational complex. It can involve multiple passes and, if necessary, may even involve human intervention. The output of the analysis, the embedding metadata, indicates the location of changeable bytes in the bitstream and the values to which these bytes can be changed. For each position listed in the embedding metadata, a number of different alternative values may be specified. At embed time, the coded payload will determine which of the multiple available values should be used for replacement. Note that in many applications, including those of interest to us, the alternative values must occupy the same number of bits in the entropy coded stream as the original values being replaced.

3. INTRA-PREDICTION BASED WATERMARKING IN H.264/AVC

3.1. Intra-Prediction mode modification for watermarking

The H.264/AVC video compression standard, like most video compression standards, achieves compression by predicting the values in a block of pixels from the values in one or more previously coded blocks of pixels. The difference between the prediction and the actual values, often called the residue, is then transform coded and quantized. The block or blocks used for the prediction, the reference blocks, can be from the same frame or from different frames. Blocks whose reference blocks come from the same frame are called Intracoded blocks or simply Intra-blocks. In this case, the prediction is often called Intra-prediction.

For the luminance samples, an entire 16×16 macroblock can be Intra-predicted as a whole or can be divided into 8×8 sub-blocks or even 4×4 sub-blocks. Each sub-block will then be Intra-predicted independently. To simplify the discussion, we only discuss 16×16 Intra-prediction. For 16×16 Intra-prediction, the following four modes are defined:

- Mode 0 (vertical) Extrapolation from samples above
- Mode 1 (horizontal) Extrapolation from samples on the left
- Mode 2 (DC) Mean of samples above and to the left
- Mode 3 (Plane) A linear plane is fitted to samples above and to the left

To decode a block, the predicted block is generated with the available pixels of previously decoded neighboring blocks as specified by the Intra-prediction mode of the current block. Then, the decoded residue pixel values are added to the predicted block. The result is the final decoded pixel block,

$$B = P + R$$

where B is the final decoded block of pixel values, P is the predicted block, and R is the block of decoded residues.

By changing the intraprediction mode of a 16×16 intracoded macroblock, the entire predicted block, P, can be changed thus changing all 256 pixels in the decoded block, B. The difference between the original block of pixel values and the watermarked block of pixel values, denoted ΔB, will be the same as the difference between the original predicted block of pixels and the watermarked predicted block of pixel values, denoted ΔP.

$$\Delta B = \Delta P \quad \text{where} \quad \Delta B = B_w - B \text{ and } \Delta P = P_w - P,$$

and the subscript 'w' indicates the watermarked version of each.

By changing the Intra-prediction mode of a macroblock, the pixels of that block will change by ΔB. For this change to be appropriate for watermarking, we require that it be detectable from the reconstructed imagery, that the change be imperceptible in the reconstructed imagery, and that the detection be robust to some predefined set of signal distortions. To this end, we evaluate the suitability of each possible change and select only those changes that meet these requirements.

The imperceptibility requirement can be interpreted in many ways. We can require that the reconstructed imagery have high visual quality or that it be indistinguishable from the original, unmarked imagery, or that the perceptibility of the changes fall below some threshold. There are many ways people judge the fidelity of a watermark. For the purposes of this paper, we allow any appropriate fidelity measure to be applied to judge whether or not, or to what extent, a proposed change meets the fidelity requirements of the application.

The requirement that the change be robustly detectable requires establishing a good feature that can be reliably measured in the decoded imagery and can be modified by changing the Intra-prediction mode of a macroblock. One feature we have investigated is the mean luminance of the macroblock. Each Intra-prediction mode change will result in a change, ΔB, in the decoded pixel values. This ΔB may have a positive or negative average value and the magnitude of the average can vary. The sign of the change can be used to encode data and the magnitude can be used as an indication of the expected robustness of the change.

A second feature that we have investigated is the variance of the reconstructed block of pixels. DC mode (mode 2) is very different from other three Intra-prediction modes in that all the 16 by 16 pixel values are predicted with a single value; the mean of the reference pixels. It is expected that an H.264/AVC encoder will use this mode when a block is smooth in nature. If we change the mode to one of the remaining three modes, the variance of the resultant block can be expected to increase. On the other hand, if an encoder chose modes 0, 1, or 3, it is expected that this block has higher fluctuation. By changing the intra-prediction mode to 2, the variance of this block will decrease. Again, the ΔB associated with a change of Intra-prediction mode may result in an increase or

decrease in the variance of the reconstructed block and the magnitude of the variance change can vary. The sign of the variance change can be used to encode data and the magnitude of the change can be used as an indication of the expected robustness of the change.

3.2. Changing Intra-prediction mode through mb_type

In H.264/AVC, the Intra-prediction mode of a 16×16 macro-block is specified in the mb_type field. The mb_type field also specifies other parameters about this block such as coded-block-pattern. Figure 4 is part of the table that lists the mb_type values with their meanings. This table, taken directly from the standard [5] , is used to find mb_type values that change the Intra-prediction mode without changing the coded-block-pattern.

mb_type	Name of mb_type	transform_size_8x8_flag	MbPartPredMode (mb_type, 0)	Intra16x16PredMode	CodedBlockPatternChroma	CodedBlockPatternLuma
0	I_NxN	0	Intra_4x4	na	Equation 7-33	Equation 7-33
0	I_NxN	1	Intra_8x8	na	Equation 7-33	Equation 7-33
1	I_16x16_0_0_0	na	Intra_16x16	0	0	0
2	I_16x16_1_0_0	na	Intra_16x16	1	0	0
3	I_16x16_2_0_0	na	Intra_16x16	2	0	0
4	I_16x16_3_0_0	na	Intra_16x16	3	0	0
9	I_16x16_0_2_0	na	Intra_16x16	0	2	0
10	I_16x16_1_2_0	na	Intra_16x16	1	2	0
11	I_16x16_2_2_0	na	Intra_16x16	2	2	0
12	I_16x16_3_2_0	na	Intra_16x16	3	2	0

Figure 4. Macroblock types for I slices [5]

For example, consider a 16×16 intracoded block with mb_type of 11. From the table, we can see that the Intra-prediction mode is 2 (DC) and that the coded-block-pattern for chroma and luma are 2 and 0 respectively. We can change the Intra prediction mode without modifying the coded block patterns, by considering mb_types 9, 10, or 12. Changing the mb_type in the stream to any other value may result in a non-compliant bitstream.

The mb_type is entropy coded in the bitstream. The two most commonly used methods for entropy coding this data are Context Adaptive Variable Length Coding (CAVLC) and Context-based Adaptive Binary Arithmetic Coding (CABAC). The current paper addresses the case in which CAVLC entropy coding is used. In this case, the mb_type is encoded with the exponential Golomb code; a variable length coding scheme. In some special applications, such as the watermarking of authored DVD disks, the coded replacement values must have exact the same length as the coded original values. In this case, we add a restriction that only mb_types whose VLC code has the same length as the VLC code of the original mb_type can be considered as a potential replacement value. Figure 5 list the bit string form and the corresponding value range of Exp-Golomb code, again taken directly from the standard [5] . From this table, we find that, using the example from above, an mb_type of 11 (fourth row of the table) would require 7 bits. The mb_type can only be replaced with another mb_type also requiring 7 bits. In this case, mb_types 9, 10, and 12 all fall in the same range (7-14) and all require 7 bits. Thus, for this example, the constant length constraint does not restrict our choices. In general, however, we can combine the VLC bit lengths of the table in Figure 5 with the macroblock types of the table in Figure 4 to deduce which mb_types can be used to replace the original mb_type by enforcing the following rules:

1. The alternative mb_type should differ only in Intra-prediction mode.

2. The length of the bit string corresponding to the alternative mb_type should be the same as that corresponding to the original mb_type (if constant bit length is required).

Bit string form	Range of codeNum
1	0
0 1 x_0	1-2
0 0 1 x_1 x_0	3-6
0 0 0 1 x_2 x_1 x_0	7-14
0 0 0 0 1 x_3 x_2 x_1 x_0	15-30
0 0 0 0 0 1 x_4 x_3 x_2 x_1 x_0	31-62
...	...

Figure 5. Ex-Golomb coding table [5]

3.3. Block selection strategy

During the off-line analysis, we find changes that satisfy the three requirements that 1) the result of the changes is a compliant bitstream, 2) the fidelity of the changes and later decoded imagery is high, and 3) the embedded data can be robustly recovered from imagery that may have been distorted after embedding. There are many ways to do this. Here we describe a simple approach in which we first build a list of all potential changes that satisfy the first requirement. We then filter that list, removing potential changes that would violate the second or third requirements. From the remaining set, we select at random as needed to embed the coded payload.

The discussion of the last section describes how we find changes that will result in a compliant stream. For each 16×16 intracoded macroblock, we identify the mb_type in the coded bitstream and list all potential alternative mb_types that change only the Intraprediction mode and are coded with the same number of bits.

Recall from our previous discussion that changing the Intraprediction mode of a macroblock has the effect of changing the pixel values by ΔB and that this change will, in turn, affect the measured detection feature of the block. Let's refer to this change in the detection feature as Δf. For each potential change on our list of compliant changes, we estimate the implied Δf. For example, if the feature is mean luminance, Δf will be the mean luminance of the ΔB block of pixels.

This list of potential changes, along with their associated values of Δf can be filtered for detectability. In the approach described here, we store the value of the feature associated with the original block of pixels in the detection metadata and make that available to the detector. For each change, we want to provide two alternative values to the embedder, one that increases the value of

the feature and one that decreases it. In other words, one alternative has a positive Δf and the other has a negative Δf. Our first pass at filtering the list of potential changes can be to remove from our list all blocks for which all Δf values have the same sign.

We next apply fidelity and robustness filters. These can be sophisticated filters that estimate the fidelity and robustness of each remaining potential change on the list and exclude changes that fail to meet some thresholds. For this example, we consider the simplest possible filters. We assume that robustness is proportional to Δf and that fidelity is inversely proportional to Δf. Thus, we limit the maximum fidelity impact by putting an upper bound on the magnitude of Δf, call it Δf_{max}, and we establish a minimum robustness by establishing a lower bound on the magnitude of Δf, Δf_{min}. Thus, in this filtering step, we remove all potential changes for which $|\Delta f|$ is outside of the range $(\Delta f_{min}, \Delta f_{max})$.

Each block must have at least two potential changes for which the $|\Delta f|$ is in the fidelity/robustness range and for which the two Δf values have opposite signs. All potential changes associated with a block that does not meet this requirement are removed from the list. For each block that meets this requirement with three potential changes that meet the fidelity/robustness requirements, two potential changes are selected and the third is discarded.

Finally, the list now contains only changes that result in a compliant marked stream, acceptable fidelity, and acceptable robustness. The block position (frame number and pixel position in the frame) along with the original value of the feature is written to the detection metadata. For the embedding metadata, we need the exact position in the bistream of the CAVLC code for the current mb_type value. As we performed the initial entropy decode, we tracked and stored these bit positions. In the embedding data, we list this position along with the CAVLC values of the two alternative mb_types. The first listed is the one that will decrease the value of the detection feature ($\Delta f<0$) and the second is the one that will increase the value of the detection feature ($\Delta f>0$).

5. WATERMARK DATA RECOVERY

Figure 3 suggested that one of the outputs from the analysis step is some detection metadata. This metadata lists the locations (frame number and pixel position) of each 16×16 block of pixels that has been changed and the original value of the feature that has been changed (e.g., mean luminance). In order to recover the embedded payload, we must therefore first recover the correct block of pixels. This may entail decoding if the imagery arrives compressed and it may require temporal and/or geometric registration. There are many approaches for obtaining this registration (e.g., [6,7,8]), but this is beyond the scope of the current paper.

Each entry in the detection metadata defines a change by its frame position and block within the frame. For each entry, the corresponding block of pixel data is extracted and from that, the detection feature is calculated. In the example above, the detection feature is the mean luminance. The calculated feature is compared to the value in the detection metadata file. The sign of the difference is calculated and the corresponding symbol is output as a symbol stream. If a data coding process was applied in the embedder then the corresponding data decoding process is applied to the symbol stream yielding the recovered payload, referred to in Figure 10 as the payload estimate.

When mean luminance is used as the detection feature, the system can become confused by global changes in brightness. In other words, a global increase in brightness can mean that the luminance extracted from the baseband imagery will always be higher than the value stored in the metadata. Thus, the symbol stream may consist of all '1' bits. To counter this, the preprocessor can add a number of reference entries into the detection metadata. A reference entry lists a frame number, block position, and original luminance value of the block for a block that is not affected by the watermarking process. The detector can read out the reference values and compare these to the corresponding values seen in the baseband imagery. The baseband imagery can then be adjusted (perhaps during registration) such that the measured luminance in the reference blocks matches that listed in the detection metadata file.

6. CONCLUSION AND NEXT STEPS

In this paper, we have described a stream replacement method for video watermarking. This method is suitable in applications in which entropy-coded, compressed video data must be modified on the fly without entropy decoding, but when pre-analysis is possible. A real-time, data byte replacement method is discussed which use embedding metadata to guide the payload embedding. Direct change of a compressed bitstream is generally difficult. In this paper, an Intra-prediction mode based method to generate the watermarking metadata is also presented.

This paper presented a method to change the H.264 encoded bitstream for watermarking. However, the whole watermarking system presented in this paper is far from complete. To achieve best robustness/fidelity, which metric to choose in the Intra-prediction based method needs further investigation. In the meantime, an efficient fidelity model needs to be incorporated into the watermarking cost analysis.

7. REFERENCES

[1] I. Cox, M. Miller, and J. Bloom: Digital Watermarking: Principles & Practice, San Mateo, CA: Morgan Kaufman, 2001.

[2] D. Simitopoulos, S.A. Tsaftaris, N.V. Boulgouris, M.G. Strintzis: Fast MPEG watermarking for copyright protection, 9th International Conference on Electronics, Circuits and Systems, 15-18 Sept. 2002 Page(s):1027 - 1030 vol.3.

[3] T. Chung, M. Hong, Y. Oh, D. Shin, S. Park: Digital watermarking for copyright protection of MPEG2 compressed video, IEEE Transactions on Consumer Electronics, Volume 44, Issue 3, Aug. 1998 Page(s):895 - 901

[4] M. Noorkami, R.M. Mersereau,: Compressed-domain video watermarking for H.264, IEEE ICIP 2005. 11-14 Sept. 2005, Volume: 2, On page(s): II- 890-3.

[5] ITU-T Recommendation H.264 | ISO/IEC 14496-10 International Standard with Amendment 1.

[6] L.G. Brown: A survey of Image Registration Techniques, ACM Computing Surveys, 24(4):325-376, 1992.

[7] B.D. Lucas and T. Kanade: An Iterative Image Registration Technique with an Application to Stereo Vision, Proceedings of the International Joint Conference on Artificial Intelligence, pp.674-679, 1980.

[8] H.S. Stone: Progressive Wavelet Correlation Using Fourier Methods, IEEE Transaction on Signal Processing, 47(1):97-107, Jan. 1999.