Towards Fast and Robust Watermarking Scheme for H.264 Video

Xun Gong and Hai-Ming Lu
Tsinghua National Laboratory for Information Science and Technology
Research Institute of Information Technology
Tsinghua University, Beijing 100084, China
gongx07@mails.tsinghua.edu.cn

Abstract

Most available H.264 watermarking methods either require large expense of time or are rather fragile to intended attacks. In this paper, we present a novel watermarking scheme, which satisfies with the requirements of both rapidity and robustness. The proposed scheme directly embeds watermarks into H.264 bitstreams at the decoder by modifying the quantized DC coefficients in luma residual blocks. To increase the robustness while maintaining the perceptual quality of the video, we employ a texture-masking-based perceptual model to adaptively choose the watermark strength for each block. A crucial problem of decoder-based watermarking algorithms is error drift, which refers to the watermark error accumulations among different blocks during intra or inter predictions. To eliminate the effects of error drift, we propose a drift compensation algorithm by estimating a compensation signal for each block before embedding the watermark bit. Experimental results show that the watermarks are highly imperceptible and resistant to common attacks. In addition, our scheme is faster and causes less increase in bit rate than previous methods.

1. Introduction

Digital watermarking technique gains increasing attentions these years for its effectiveness in protecting multimedia copyrights. In the literature, numerous video watermarking algorithms based on previous standards such as MPEG-2 and H.263 have been proposed. These methods however cannot be directly applied to H.264/AVC [1] due to the employment of many new features in H.264. As H.264 achieves a higher compression efficiency and is expected to replace previous standards within the next few years, developing watermarking schemes appropriate for it is greatly desired.

Recently, a few papers began to investigate H.264 watermarking algorithms. In [2], the authors present a blind watermarking algorithm which embeds the watermark information into the relations between predicted DCT coefficients. Their method is robust to compression attacks with high compression ratios, but requires fully decompressing the video before watermark embedding and compressing the watermarked sequence afterward. For real-time applications, faster schemes like approaches in [3, 4] are more appropriate. The authors in [3] propose a low-complexity scheme which embeds watermarks into the quantized AC coefficients of I-frame macroblocks. In [4], the authors embed the watermark bits by mapping eligible CAVLC codes to the unused code space. Both methods need only partial decompression or even no decoding, but are too fragile under intended attacks such as re-encoding. In order to enhance the robustness, the authors in [5] employ a 4×4 DCT block-based human visual model, and embed the watermarks adaptively into quantized AC coefficients of the luma residual blocks. However, their algorithm is only suitable for encoder-based watermarking schemes, which means complete decompressions are still required. When embedding the watermarks at the decoder, it suffers from severe degradations caused by the error drift problem.

The goal of this paper is to propose a H.264 watermarking scheme, that is both fast and robust enough for practical applications. In order to be fast, a decoder-based scheme is chosen. To ensure the watermarks' transparency and robustness, we adaptively adjust the watermark signal based on a texture-based perceptual model and propose a drift compensation algorithm. The rest of this paper is organized as follows. In Section 2, we describe details about the proposed watermarking scheme. Simulation results and analysis are given in Section 3. In final, Section 4 concludes our work.

2. The Proposed Scheme

2.1. The 4×4 DCT Block Classification

According to the texture masking property of human visual system, the human eye is more sensitive to errors in

978-0-7695-3454-1/08 $25.00 © 2008 IEEE
DOI 10.1109/ISM.2008.16

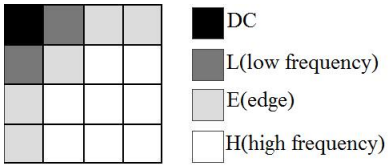


Figure 1. Classification indicators in a 4×4 DCT block

plain areas than in highly textured areas, which means embedding watermarks in plain blocks causes visible artifacts more easily than in texture blocks. Thus, to increase the watermark robustness while preserving the visual quality of the video, the watermark strength in each block should be chosen individually based on the block texture level. To this end, we first classify all 4×4 luma blocks into *plain*, *edge* and *texture* types, and then decide the amplitude of watermark signal according to the block type. The key of our classification method is a 4×4 DCT perceptual model, adapted from the 8×8 DCT perceptual model in [6], where it is originally proposed to design an efficient JPEG encoder. The steps of classification process are provided as follows:

S1: Divide all coefficients in a 4×4 DCT block into four groups as marked with different colors in Fig. 1.

S2: Calculate the absolute sum of the coefficients in each group, denoted as DC, L, E and H respectively.

S3: Use the values of $E + H$, $\frac{L}{E}$ and $\frac{L+E}{H}$ as classification indicators, and compare them with a series of experimentally determined thresholds to obtain the exact block type. In general, a block with large values of $\frac{L}{E}$ and $\frac{L+E}{H}$ is highly possible to be on edge area, while a small value of $E + H$ always indicates a plain block. More details about the comparison procedure are given in [6].

After the block type is obtained, we can choose a proper watermark intensity for the block. Normally, the appropriate watermark intensity for different block types increases in the order of plain, edge, texture. By adaptively adjusting the watermark amplitudes, we can achieve a controllable tradeoff between the robustness of the embedded watermarks and the perceptual quality of the watermarked video.

2.2. Drift Compensation

Like other hybrid coding standards, H.264 uses intra and inter prediction algorithms to reduce both spatial and temporal redundancies. The predicted samples in each block are calculated using reference samples in adjacent blocks or from previous frames. Therefore, the effects of watermarks embedded to the residuals at the decoder would accumulate among different blocks when reconstructing the video content from residual blocks. If the error accumulations are

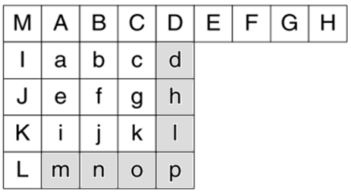


Figure 2. The predicted and reference samples in I4MB type

severe enough to cause visible artifacts, then *error drift* occurs. Note that only the I-frames are watermarked in our scheme, the main reason for the error drift problem is intra prediction.

H.264 employs a new intra prediction algorithm, which provides two prediction types for luma blocks: I4MB and I16MB [1]. We only discuss I4MB type here, which contains 9 prediction modes in total. Each mode represents a prediction direction and corresponds to a distinct prediction formula. As illustrated in Fig. 2, the predicted samples a-p are obtained by calculating a weighted mean of the reference samples A-M using the prediction formula according to a selected prediction mode. For instance, the predicted samples are all predicted as (A+B+C+D+I+J+K+L)/8 if the selected mode is 2 (DC prediction). When any of A-M is modified after watermarking, the watermark error may propagate to a-p, too. We define the watermark error obtained from adjacent blocks as *propagating error*, denoted as Δ, and the final sample modification after watermarking as *expected alternation*, denoted as ξ. To eliminate the effects of error drift, we need to estimate a compensation signal and subtract it from the original watermark signal before the embedding process. The steps of compensation procedure are provided as follows:

S1: Calculate the average propagating error in the block as given by

$$\Delta_{avg} = \frac{\Delta_a + \Delta_b + \cdots + \Delta_p}{16} \tag{1}$$

The sample propagating error Δ_a-Δ_p is obtained from the expected alternations in adjacent blocks using the selected prediction formula. For instance, Δ_a can be calculated as

$$\Delta_a = pred_a(\xi_A, \xi_B, \cdots, \xi_M, mod) \tag{2}$$

where $pred_a$ is the prediction formula set of sample a, and mod is the selected prediction mode of current block. ξ_A-ξ_M are calculated and recorded already during the drift compensations in reference blocks.

S2: Calculate the corresponding propagating error in the quantized DC coefficient (the compensation signal) as

$$\Delta_{DC} = \frac{\Delta_{avg}}{\Delta_0} \tag{3}$$

Δ_0 is the alternation in each sample when the quantized DC coefficient of current block is modified by 1, which can be calculated as

$$\Delta_0 = C^T \left(\begin{bmatrix} Q & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \otimes E \right) C \quad (4)$$

where Q is the quantization step size, C is the inverse transform matrix, and E is a matrix of scaling factors, given in [1]. Normally, the value of Δ_0 is $0.25Q$.

S3: Subtract Δ_{DC} from the original watermark signal γ to obtain the final watermark signal γ' as

$$\gamma' = \gamma - \Delta_{DC} \quad (5)$$

S4: Calculate and record the expected alternations in current block for drift compensations in subsequent blocks. We only need to calculate the alternations in 7 samples as marked in Fig. 2, for these samples will be used in future predictions. For instance, the expected alternation ξ_m is calculated as

$$\xi_m = \gamma' \cdot \Delta_0 + \Delta_m \quad (6)$$

2.3. Embedding Process

Fig. 3 illustrates the embedding procedure, which contains following steps:

S1: Read encoding parameters and the residual block data from the original bitstream. Decode each 4×4 luma residual block partially as quantized coefficients.

S2: Obtain the block type using DCT coefficients of current residual block together with previously predicted blocks, and choose the watermark intensity for current block accordingly. Use a bipolar sequence $W \in \{-1, 1\}$ with zero mean and variance one to obtain the original watermark signal as

$$\gamma_i = I(b) \cdot w_i \quad (7)$$

where w_i is the *i*th bit in the watermark sequence, and $I(b)$ is the watermark intensity decided based on current block type b.

S3: Perform drift compensation process using the prediction mode of current block to obtain the final watermark signal γ'. Modify the quantized DC coefficient as

$$DC_i' = DC_i + \gamma_i' \quad (8)$$

S4: Encode the modified residual block to produce the watermarked bitstream.

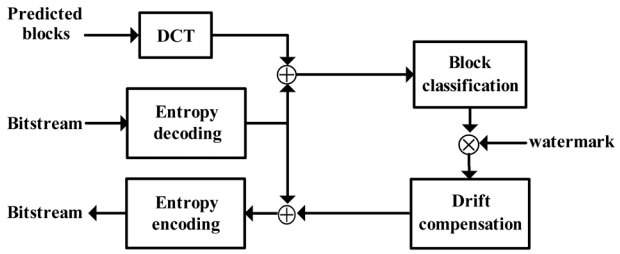


Figure 3. Watermark embedding procedure

2.4. Detection Process

To ensure the robustness under re-encoding attacks, the watermarks are detected on picture level in our scheme. In the following part, we denote the original watermark sequence used in the embedding process as W, and the watermark sequence actually embedded in the test video as W^*. The steps of detection procedure are given as follows:

S1: Decode the test bitstream into picture frames and carry out 4×4 block DCT on all I-frames.

S2: Perform the block classification procedure to locate embedded blocks (with nonzero intensity). Since only the DC coefficients are modified in our scheme, we can obtain the correct block type from AC coefficients in the watermarked frames using the classification method introduced in Section 2.1.

S3: Calculate a detection statistic μ as

$$\mu = 4 \times \frac{\sum_{i=1}^{n} mean_i \cdot w_i}{n} \quad (9)$$

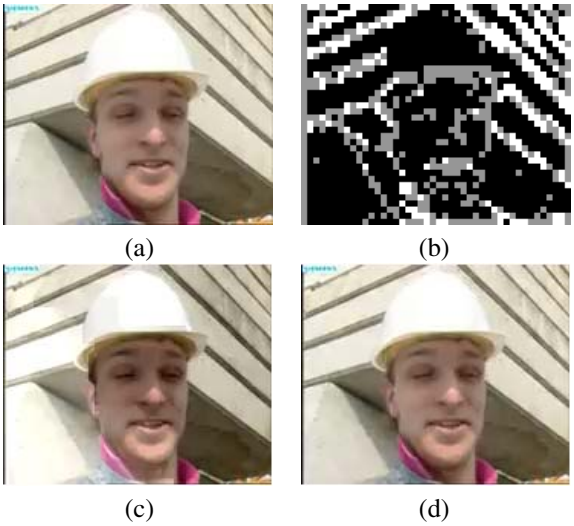
where n is the total number of watermark bits, and $mean_i$ is the mean value of luma samples in the *i*th embedded block. As the quantized DC coefficient has a linear relationship with the luma block mean, Eq. (9) can be rewritten as

$$\mu = 4 \times \frac{\sum_{i=1}^{n} mean_{0i} \cdot w_i}{n} + Q \cdot \frac{\sum_{i=1}^{n} I(b) \cdot w_i^* \cdot w_i}{n} \quad (10)$$

where $mean_{0i}$ is the mean value of the *i*th block before watermarking. Note that W is independent of the video content and has zero mean, the first term in Eq. (10) is supposed to be near zero when n is large enough. In addition, if the watermark intensity uses a single value 1, we have

$$\mu = \begin{cases} Q & if\ W^* = W \\ 0 & if\ W^* \neq W\ or\ W^* = 0 \end{cases} \quad (11)$$

S4: Choose a proper detection threshold between 0 and Q (e.g., $Q/2$). If μ is lager than the detection threshold, then the test video contains the original watermark. Otherwise, the original watermark does not exist.



(a) (b) (c) (d)

Figure 4. Compensation evaluation (a) the original I-frame (b) the block type map (c) watermarked with no drift compensation, PSNR=19.78dB (d) watermarked with drift compensation, PSNR=40.40dB

3. Experimental Results

Our scheme is implemented at the decoder of JM 11.0 [7]. The test bitstreams are 4 standard QCIF sequences encoded with a fixed quantization step size Q=16 (the quantization parameter is 28) and an intra period of one. Each test sequence has 100 I-frames. Only IM4B type is enabled for intra prediction in our tests. The watermark intensity is set as 0 for all plain blocks and 1 for edge and texture blocks.

3.1. Effectiveness of Drift Compensation

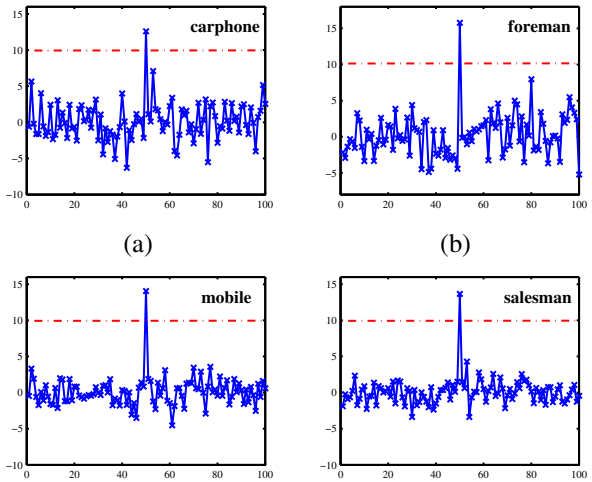
Fig. 4 presents the results of an I-frame from *foreman* sequence watermarked with and without drift compensation respectively. The embedded blocks are edge and texture types as marked in grey and white in Fig. 4 (b). It is observed from Fig. 4 (a) and Fig. 4 (d) that the watermarks in our scheme cause no visible degradations to the video picture. We can evaluate the effectiveness of the proposed drift compensation algorithm by comparing Fig. 4 (c) with Fig. 4 (d). It is clear that the watermarks cause obvious artifacts in the frame watermarked without compensations, while no such artifacts can be found when drift compensation is employed. Besides, the PSNR difference between two cases also indicates the effectiveness of drift compensation.

3.2. Embedding and Detection Results

We compare the embedding performance of our scheme with the algorithm in [5], which is implemented at the encoder of JM 11.0 in our experiments. Fig. 5 presents the results of an I-frame from *carphone* sequence watermarked with both methods. It shows that the visual qualities in two cases are comparable. Table 1 lists more results of the embedding performances, including watermark capacity, bit rate increase, time cost and the PSNR values. The time cost is calculated by dividing the total run time with the I-frame number, and the PSNR value is compared between original and watermarked I-frames after decoding. Although the capacities are comparable, it is observed that our scheme outperforms [5] in compression efficiency and time saving. On average, our scheme saves about 45% storage and 96% time expense compared with the algorithm in [5].

(a) (b)

Figure 5. Visual quality comparison between our scheme and the method in [5] (a) watermarked I-frame in our scheme (b) watermarked I-frame from [5]



(a) (b) (c) (d)

Figure 6. Watermark detection performances

We use 100 bipolar sequences with normal distribution in the detection performance tests. The 50th sequence is our original watermark. The detection results of all sequences are illustrated in Fig. 6, where the vertical axis represents the value of the detection statistic μ. When the detection threshold is set as 10 (the dashed lines in Fig. 6), it is observed that the original watermarks are correctly detected because the middle peaks are all above the dashed lines.

Table 1. Embedding performances comparison between our scheme and [5]

Bitstream	Average number of watermark bits in an I-frame		Bit rate increase (%)		Time (ms)		$PSNR_{avg}$ (dB)	
	Our scheme	Method in [5]	Our scheme	Method in [5]	Our scheme	Method in [5]	Our scheme	Method in [5]
carphone	500	507	3.28	4.38	72.19	2274.86	41.24	38.32
foreman	597	591	2.64	4.62	75.56	2349.58	40.18	37.28
mobile	1219	1225	0.68	2.17	90.94	2717.32	37.22	34.20
salesman	750	736	2.49	4.43	74.30	2554.63	39.57	36.69

Table 2. Robustness to adding noise

Bitstream		no attack	Gaussian noise σ^2=19	Gaussian noise σ^2=480
carphone	μ	12.60	12.60	12.62
	μ'_{avg}	0.04	0.04	0.04
	μ'_{max}	7.10	7.08	7.02
	μ'_{min}	-6.31	-6.28	-6.41
foreman	μ	15.76	15.75	15.63
	μ'_{avg}	0.10	0.10	0.11
	μ'_{max}	7.96	7.99	7.87
	μ'_{min}	-5.24	-5.31	-5.15
mobile	μ	14.06	14.05	13.71
	μ'_{avg}	0.009	0.01	-0.01
	μ'_{max}	3.57	3.56	3.63
	μ'_{min}	-4.53	-4.52	-4.60
salesman	μ	13.67	13.66	13.63
	μ'_{avg}	-0.003	-0.004	-0.01
	μ'_{max}	4.29	4.30	4.38
	μ'_{min}	-3.38	-3.41	-3.38

Table 3. Robustness to requantization

Bitstream		QP=24	QP=32	QP=36
carphone	μ	11.73	10.53	9.64
	μ'_{avg}	0.04	0.05	0.04
	μ'_{max}	7.09	7.10	7.06
	μ'_{min}	-6.29	-6.27	-6.33
foreman	μ	15.07	13.93	13.17
	μ'_{avg}	0.10	0.10	0.10
	μ'_{max}	7.94	7.96	8.04
	μ'_{min}	-5.24	-5.22	-5.22
mobile	μ	13.72	12.95	12.31
	μ'_{avg}	0.009	0.009	0.01
	μ'_{max}	3.59	3.60	3.63
	μ'_{min}	-4.53	-4.50	-4.53
salesman	μ	12.90	11.54	10.60
	μ'_{avg}	-0.005	-0.007	-0.005
	μ'_{max}	4.29	4.23	4.22
	μ'_{min}	-3.40	-3.40	-3.38

3.3. Results of Robustness Tests

To evaluate the robustness of our watermarks, we employ Gaussian noises and recompression with different quantization step sizes as intended attacks. The test results are presented in Table 2 and Table 3, where σ^2 is the variance of Gaussian noise, μ is calculated with the original watermark sequence, and μ' is calculated with the rest 99 sequences. Notice that μ'_{avg} is near 0 and μ is close to 16 when no attack is applied. This exactly agrees with our theoretic analysis in Eq. (11). When the watermarked videos are attacked by adding noises or re-encoded with different step sizes, it is observed that μ and μ' are still far apart enough for correct detections, which demonstrates that the watermarks in our scheme are resistant to these two kinds of attacks.

4. Conclusion

A novel decoder-based watermarking scheme for H.264 is proposed. The watermarks are adaptively embedded into luma residual blocks after partially decoding the bitstream. A compensation algorithm is proposed to solve the error drift problem in decoder-based schemes. Experimental results reveal that the proposed scheme can achieve enough robustness while preserving the video's visual quality. In comparison with previous methods, our scheme is faster and the loss in compression efficiency is less.

References

[1] I. E. G. Richardson, *H.264 and MPEG-4 Video Compression: Video Coding for Next generation Multimedia*. England: John Wiley & Sons, 2003.

[2] G. Z. Wu and Y. J. Wang, "Robust watermark embedding/detection algorithm for H.264 video", *Journal of Electronic Imaging*, vol. 14, pp. 013013-1-9, January, 2005.

[3] M. Noorkami and R. M. Mersereau, "Compressed-domain video watermarking for H.264", *ICIP 2005*, pp. II-890-3, 2005.

[4] B. G. Mobasseri and Y. N. Raikar, "Authentication of H.264 streams by direct watermarking of CAVLC blocks", *Proceedings of SPIE*, pp. 65051W-1-5, 2007.

[5] M. Noorkami and R. M. Mersereau, "A framework for robust watermarking of H.264-encoded video with controllable detection performance", *IEEE transaction on forensics and security*, vol. 2, no. 1, pp. 14-23, December, 2007.

[6] H. H. Y. Tong and A. N. Venetsanopoulos, "A perceptual model for JPEG application on block classification, texture masking, and luminance masking", *ICIP 98*, pp. 428-432, 1998.

[7] H.264 Reference Software Group. [Oline]. Available: http://iphome.hhi.de/suehring/tml/.