Cover Page

1) Title of the paper:

Robust drift-free bitrate preserving H.264 watermarking

2) authors' affiliation and address:

LUNAM Université, Université de Nantes IRCCyN UMR CNRS 6597
Polytech Nantes, rue Christian Pauc
BP 50609, 44306 Nantes, France

Tel.: +33 240683052
Fax: +33 240683232

3) e_mail address:

Florent.Autrusseau@polytech.univ-nantes.fr

4) Journal & Publisher information:

Springer
http://www.springer.com/computer/information+systems+and +applications/journal/530

5) bibtex entry:

```
@article{SpringerMMS2013,
  author = {W. Chen and Z. Shahid and T. Stuetz and F.
Autrusseau and P. Le Callet},
  title = {Robust drift-free open-loop H.264 watermarking},
  journal = {accepted for publication in Springer, Multimedia
Systems},
  year = {2013},
  volume = {Special Issue on Digital Media Security and Right
Management},
  number = {ISSN 0942-4962}
}
```

Noname manuscript No.
(will be inserted by the editor)

Robust drift-free bitrate preserving H.264 watermarking

Wei Chen · Zafar Shahid · Thomas Stütz · Florent Autrusseau · Patrick Le Callet

Received: date / Accepted: date

Abstract This paper presents a novel method for open-loop watermarking of H.264/AVC bitstreams. Existing watermarking algorithms designed for previous encoders, such as MPEG-2 cannot be directly applied to H.264/AVC, as H.264/AVC implements numerous new features that were not considered in previous coders. In contrast to previous watermarking techniques for H.264/AVC bitstreams, which embed the information after the reconstruction loop and perform drift-compensation, we propose a new completely intra-drift- free watermarking algorithm. The major design goals of this novel H.264/AVC watermarking algorithm are: runtime-efficiency, high perceptual quality, (almost) no bit-rate increase and robustness to re-compression. The watermark is extremely runtime-efficiently embedded in the compressed domain after the reconstruction loop, i.e., all prediction results are reused. Nevertheless intra-

LUNAM Université, Université de Nantes
IRCCyN UMR CNRS 6597
Polytech Nantes, rue Christian Pauc
BP 50609, 44306 Nantes, France
Tel.: +33 240683052
Fax: +33 240683232
E-mail: Florent.Autrusseau@univ-nantes.fr

drift is avoided, as the watermark is embedded in such a way that the pixels used for the prediction are kept unchanged. Thus there is no drift as the pixels being used in the intra prediction process of H.264/AVC are not modified. For watermark detection, we use a two stages cross-correlation. Our simulation results confirm that the proposed technique is robust against re-encoding and shows a negligible impact on both the bit-rate and the visual quality.

Keywords Video watermarking · H.264 · perceptual weighting · bit-rate preservation

1 Introduction

H.264/AVC is a state-of-the-art video compression standard and has become the most widely deployed video codec in almost all applications: video distribution in the Internet as well as on Blu-ray discs, even mobile devices employ H.264/AVC to compress captured video data. The ubiquity of digital content and its ease of duplication and modification calls for technical solutions for copyright protection and authentication; digital watermarking is an integral part of such solutions. As digital video content is mostly coded with H.264/AVC, watermarking solutions that are tailored to the specifics of this standard can greatly improve the performance of digital watermarking systems, e.g., with respect to runtime performance.

In H.264/AVC, the watermark could be embedded at the various stages in the compression pipeline, i.e., pre-compression or spatial domain, transform coding stage, QTCs (quantized transform coefficients) stage and entropy coding stage. So, basically, the work on video watermarking can be divided into two main approaches: 1) closed-loop watermarking, i.e., watermark em-

bedding requires to consider the coder reconstruction loop (any modification due to watermark embedding might likely affect further encoding of information). 2) open-loop watermarking, i.e., the watermark embedding modifies only a well defined set of information without affecting the encoding process of any other information. The closed-loop watermarking almost needs complete re-compression while open-loop watermarking embeds a watermark in compressed domain (syntax elements before entropy encoding) or bitstream domain (bit substitutions of the final bitstream). Thus open-loop watermarking either requires only the entropy encoding stage or no stages from the compression. Therefore open-loop watermarking is extremely runtime-efficient compared to closed-loop watermarking. In Section 2.1, we present the recent research on closed-loop watermarking, while open-loop watermarking approaches are discussed in Section 2.2. Embedding within the reconstruction loop avoids drift, shows a small impact on rate distortion (RD) performance and offers a high watermark payload. It also has the significant disadvantage that it requires re-computation of all prediction decisions for the embedding of each single watermarking message [25]. Thus it requires the computationally most complex parts of video compression and therefore cannot be used for applications with runtime constraints or in applications which require numerous watermark messages to be embedded. Active fingerprinting is an example which usually requires numerous watermarks, as each buyer's fingerprinting code is embedded in the visual content to detect the traitor in case of illegal distribution of the content.

On the other hand, if the watermark embedding occurs after the reconstruction loop (open-loop), the most complex parts of the compression pipeline are omitted. Watermark embedding modifies only prediction residuals on the

syntax level. In open-loop watermarking, prediction is performed from watermarked content on the decoder side and from the original content on the encoder side. Thus there is a mismatch between encoder and decoder side predictors, which accumulates over time. This accumulating mismatch is referred as drift and makes open-loop watermarking very challenging, as visual quality constraints are hard to meet. As video data are usually stored and distributed in a compressed format (very often H.264), it is often impractical to first decode the video sequence, embed the watermark and then re-compress it. A low-complexity video watermarking solution requires that the embedding be conducted in an open-loop fashion. Furthermore common design goals of watermarking algorithms should also be met, such as robustness and imperceptibility. In this paper, we present an algorithm for open-loop watermark embedding in H.264/AVC bitstreams, which is robust against re-compression. At the same time, the embedding is imperceptible and avoids intra-drift, by keeping the predictor pixels unchanged.

In previous work we presented the basic idea for intra-drift-free H.264/AVC bitstream watermarking in [4], but could only report very preliminary results, which employed ad-hoc solutions of the system of linear equations and used a constant embedding strength without any human visual system model. In this paper, we present a significantly improved version of our basic idea; we now select the most suitable patterns from the complete set of solutions, and use the DCTune perceptual model [26] to modulate the watermark embedding strength. The use of DCTune improves the basic approach in two ways: First, visible artifacts are avoided in the watermarked video. Second, as the watermark embedding strength is increased to the DCTune imperceptibility

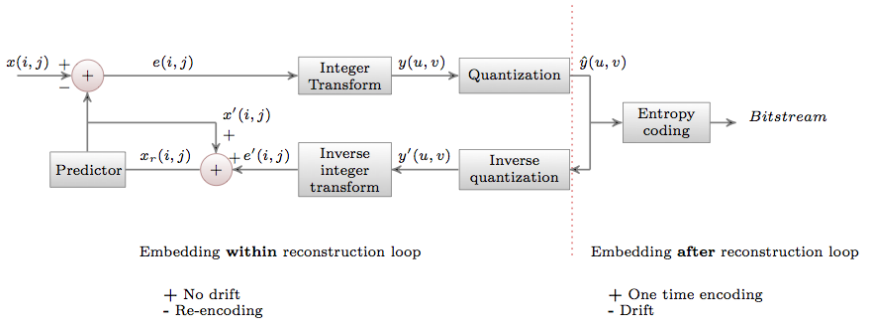


Fig. 1: Two main categories of watermarking in video codec: 1) During the encoding process or spatial domain, 2) In the bitstream domain.

limit, the improved technique offers higher robustness. Furthermore, a comprehensive analysis and a comprehensive evaluation of our proposed technique is performed in this paper.

The remainder of the paper is organized as follows: Recent work on H.264/AVC watermarking is presented in Section 2. The proposed technique including the derivation of the linear system, watermarking embedding and detection are presented in Section 3. Experimental results are presented in Section 4. In Section 5 we discuss the performance and possible improvements of the presented algorithm. Final conclusion are drawn in Section 6.

2 Recent work

For video content, watermarking is often strongly tied to the compression. As previously explained, two distinct approaches can be used for video watermarking, closed-loop watermarking, where the information is embedded either before or during compression, or open-loop watermarking, where the watermark is embedded during the entropy coding or directly within the bitstream. Figure 2 summarizes the different embedding stages that we can encounter in

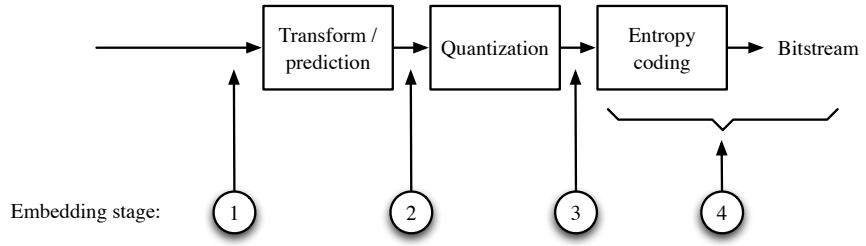


Fig. 2: Classification of watermarking schemes on the basis of working domain: (1) Pre-compression, (2) Transform domain, (3) Quantized transform domain, (4) Open-loop.

the literature, these stages are numbered within the circles. In the following subsections we will describe state-of-the-art methods for these two approaches.

2.1 Closed-loop watermarking

Embedding before or during compression can be subdivided into three main classes namely, embedding before compression, embedding in transform coefficients, and embedding in QTCs (quantized transform coefficients):

In the pre-compression stage, denoted stage 1 in Figure 2, watermarking can be performed in the pixel domain [5,3] or more commonly in a transform domain like DFT [11,6] or DWT [27]. Pre-compression watermarking approaches can also be performed on multiple frames [3,6]. In [23], Pröfrock *et al.* have presented a watermarking technique in the spatial domain, robust to H.264/AVC compression attacks with more than 40:1 compression ratio. The watermark is only contained in *intra* frames similar to the approach presented in [30]. The watermark can also be embedded in the transformed coefficients before quantization as proposed by Golikeri *et al.* [7] (see stage 2 in Figure 2). Visual models developed by Watson [26] were employed.

Some algorithms [8,21,30,25] embed the watermark in QTCs of H.264/AVC (see stage 3 in Figure 2). Noorkami and Merserau have presented a technique to embed a watermark message in both *intra* and *inter* frames in all non-zero QTCs [21]. They claimed that visual quality of *inter* frames is not compromised even if the message is embedded only in non-zero QTCs. In [8], Gong and Lu embedded watermarks in H.264/AVC video by modifying the quantized DC coefficients in *luma* residual blocks. To increase the robustness while maintaining the perceptual quality of the video, a texture-masking-based perceptual model was used to adaptively choose the watermark strength for each block. To eliminate the effects of drift, a drift compensation algorithm was proposed which adds a drift compensation signal before embedding the watermark.

In [25], Shahid *et al.* embedded watermarks in H.264/AVC video by modifying the quantized AC coefficients in *luma* and *chroma* residual blocks. This method embeds a watermark message inside the reconstruction loop to avoid drift, and hence needs re-compression for each watermark message. This algorithm has a negligible compromise on RD performance and is useful for high payload metadata hiding. While only modifying the AC QTCs above a certain threshold, the scheme offers 195 kbps payload with 4.6% increase in bit-rate and 1.38 dB decrease in PSNR at QP value of 18.

2.2 Open-loop watermarking

To avoid decoding followed by the extremely computationally demanding re-compression combined with watermarking, some methods have suggested embedding watermark messages in an open-loop fashion, e.g., [12,13,18,19,28]

to achieve low-complexity video watermarking (see stage 4 in Figure 2). In [1], perceptual models are adopted to mitigate the visual artifacts. In [14], a compressed domain watermarking technique called differential energy watermark (DEW) is proposed. In [15], the authors proposed a watermarking algorithm named different number watermarking (DNW) algorithm, embedding the mark in the texture and edges. For all of the above techniques, the main problems are visible artifacts and bit-rate increase.

In previous work the addition of a drift compensation signal to control / eliminate the drift, i.e., visible artifacts, has been proposed [10, 9]. In [10], Huo *et al.* have proposed to process only a subset of the transformed coefficients as opposed to process all the coefficients as suggested in the other compensation methods. Kapotas *et al.* [12] have presented a data hiding method in H.264/AVC streams for fragile watermarking. It takes advantage of the different block sizes used by the H.264 encoder during the *inter* prediction stage in order to hide the desirable data. The message can be extracted directly from the encoded stream without the need of the original host video. This approach can be mainly used for content-based authentication. In [20], the authors propose a self-collusion resistant watermarking embedding algorithm using a key dependent embedding strategy. Their algorithm is reported to cause only one percent increment of the bit-rate; however, it is not robust to recompression. In [19], the same authors present a new algorithm that is more robust, but the increment of bit-rate is up to 4-5%. In [24], Qiu *et al.* propose a hybrid watermarking scheme that embeds a robust watermark in the DCT domain and a fragile watermark in the motion vectors in the compressed domain. This approach is not robust against common watermarking attacks.

Watermark embedding may be conducted directly on the bitstream as well, i.e., by substitution of certain bits. In [18], authentication of H.264/AVC is performed by direct watermarking of CAVLC codes. In [13], Kim *et al.* present a new algorithm embedding the watermark in the sign bit of the *trailing ones* in CAVLC of H.264/AVC with no change in bit-rate with PSNR higher than 43 dB. But, this technique is not robust against attacks such as re-compression with different encoding parameters and common signal processing attacks. Zou and Bloom [28] have proposed to perform direct replacement of CAVLC for watermarking purpose. For CABAC [29], they have proposed a robust watermarking method using Spread Spectrum modulation while still preserving the bit-rate. This technique is robust against slight shift, moderate downscaling and re-compression. To keep the watermark artifacts invisible and to ensure the robustness, fidelity and robustness filters were used. In [16], the authors present an algorithm for watermarking *intra* frames without any drift. They propose to exploit several paired-coefficients of a 4×4 DCT block to accumulate the embedding induced distortion. The directions of intra-frame prediction are used to avert the distortion drift. The proposed algorithm has high embedding capacity and low visual distortions. Since this technique is based on intra prediction direction, it is not robust against re-encoding attack, which is the most common non-intentional attack.

Open-loop video watermarking algorithms face major challenges / limitations: First, the payload of such algorithms is significantly reduced, i.e., up to a few bytes per second as explained in [9]. Second, if drift compensation is not applied there is a continuous drift, which significantly distorts the visual quality. Third, if drift compensation is applied, the algorithms face considerable bit-rate increases. Fourth, the robustness is often very limited.

In [9] and [32], the authors emphasize the importance of bit-rate preserving watermarking algorithms.

The goal of this paper is to present a robust open-loop watermarking algorithm for H.264/AVC with a minimal increment of the bit-rate and robustness against recompression. To achieve these goals, drift is avoided instead of compensated which allows to almost preserve the original bit-rate as no compensation signals need to be coded.

3 Drift-free open-loop H.264 watermarking

The main challenge in the design of an H.264/AVC open-loop watermarking algorithm is drift, which is a result of *intra* and *inter* prediction [20, 19]. If the watermark embedding algorithm modifies the pixels which are used in these processes, the result is a drift error which accumulates, i.e., multiple drift errors add up to induce clearly visible distortions. In the proposed watermarking technique, a watermark message is embedded in the transformed domain, while making sure that only those pixels are modified which are not used for the intra prediction of future blocks. Capital letters refer to matrices, e.g., C, Y, W and lower case letters with indices refer to the elements of a matrix, e.g., x_{ij} represents jth element in ith row of matrix X.

The embedding of the watermark modifies the original pixel value x_{ij} by adding the watermark signal w_{ij}, which results in the watermarked signal x^w_{ij}.

$$x^w_{ij} = x_{ij} + w_{ij} \tag{1}$$

The values y_{ij} are the pixels in the next block, which is predicted from x_{ij}. A pixel value y_{ij} can be decomposed into its predictor p_{ij} and residual r_{ij}:

$$y_{ij} = p_{ij} + r_{ij} \tag{2}$$

At the encoder side, p_{ij} is derived from x_{ij}, but after the watermark embedding, x_{ij} changes into x_{ij}^{w}, so p_{ij} changes to p_{ij}^{w}, y_{ij} will thus become $y_{ij}^{'}$;

$$y_{ij}^{'} = p_{ij}^{w} + r_{ij} \tag{3}$$

Thus the drift error e is:

$$e \;=\; y_{ij}^{'} - y_{ij} \tag{4}$$

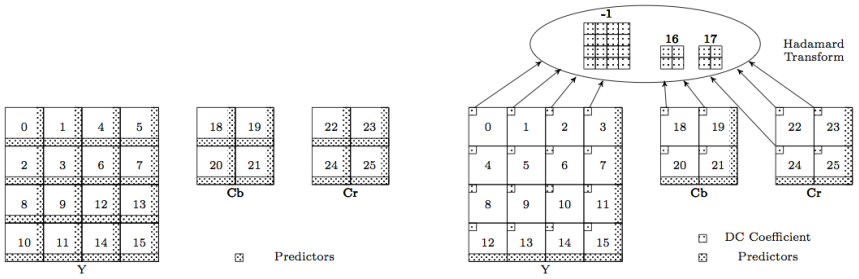
In this work, we completely avoid this drift error, as the reference pixel values for the *intra* prediction remain unchanged.

$$p_{ij}^{w} = p_{ij}, \qquad y_{ij}^{'} = y_{ij}, \qquad \Rightarrow \qquad e \;=\; 0. \tag{5}$$

In Section 3.1, we present the solutions of a system of linear equations, which allow to embed a watermark message in the DCT coefficients without modifying boundary pixels which are used for prediction of future blocks. The watermark embedding step is explained in Section 3.2, while watermark detection is presented in Section 3.3.

3.1 Intra-drift-free DCT-watermarking: solutions of a system of linear equations

In the spatial prediction of H.264, the pixels of a macroblock (MB) are predicted by using only information of already transmitted blocks in the same



(a) $I_{4\times4}$ mode (b) $I_{16\times16}$ mode

Fig. 3: H.264/AVC *intra* prediction in spatial domain: (a) Prediction is performed from pixels at left and top of every 4×4 block for $I_{4\times4}$ mode. Moreover, 4×4 blocks are transmitted in a dyadic manner inside a MB, (b) Prediction is performed at MB level for $I_{16\times16}$ mode. The order of transmission of 4×4 blocks inside a MB is in a raster scan fashion. In this mode, DC coefficients are further transformed using Hadamard transform and are sent before AC coefficients.

video frame. In H.264/AVC, two types of spatial predictions are available: $I_{4\times4}$ and $I_{16\times16}$. The $I_{4\times4}$ mode is based on predicting each 4×4 *luma* block separately. Pixels which are used for prediction in this mode are shown in Figure 3(a). Nine prediction modes are available and it is well suited for coding the detailed areas of the frame. The $I_{16\times16}$ mode, on the other hand, performs prediction of the whole 16×16 *luma* block. Figure 3(b) shows the pixels that are used for prediction in this mode. In this mode, DC coefficients are further transformed using Hadamard transform and are sent before AC coefficients. It is more suited for coding the smooth areas of the frame and has four prediction modes. In this paper, we are presenting the algorithm for a watermark embedding in the 4×4 *luma* blocks of *intra* frames. Figure 4 shows the nine different predictions modes for a 4×4 block size.

Figure 4(j), shows the labeling of prediction sample. The prediction is based on pixels labeled A to M for the current block. Thus, the d, h, l, p, m, n, o pixels will be used for future prediction. Since it is only the residual r_{ij} which

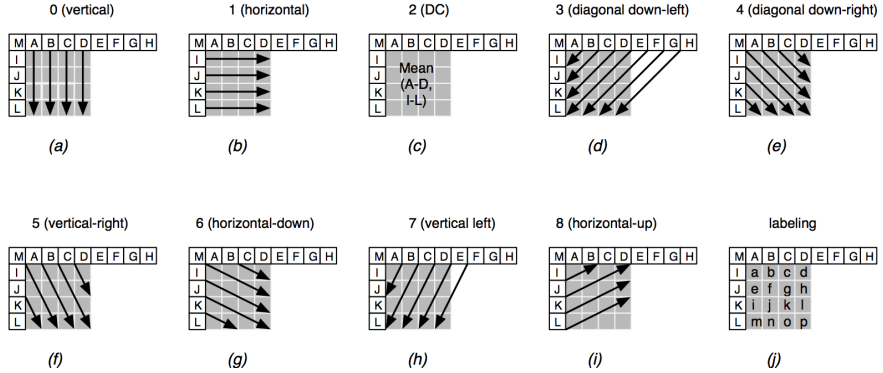


Fig. 4: Nine *intra* prediction modes (sub figures a to i) and labeling of the pixels (j).

is coded in the bitstream and is watermarked in bitstream domain watermarking techniques, if the residual r_{ij} for pixels at positions d, h, l, p, m, n, o in Figure 4(j) are unchanged, then the original pixels will remain unchanged. Since these pixels are used in the prediction, the error brought by the watermark embedding process will not propagate in the *intra* frame. Moreover, if the prediction mode is unchanged, the bit-rate increment will only come from the entropy coding of a few modified coefficients and thus remain low. Since we aim to embed the watermark in the residual in the compressed domain, we need to explore the relationship between the spatial domain pixel values and the compressed domain coefficients.

The 4×4 residual matrix X is derived by the 4×4 de-quantized transform coefficient matrix Y as follows:

$$X = C^T Y C \tag{6}$$

The matrix C is defined in the H.264 standard.

$$C = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0.5 & -0.5 & -1 \\ 1 & -1 & -1 & 1 \\ 0.5 & -1 & 1 & -0.5 \end{pmatrix} \tag{7}$$

Transform matrix Y will be modified to Y', such that the border pixels of $X' = C^T Y' C$ are the same as those in X, i.e.:

$$C^T Y' C - C^T Y C = X' - X = D = \begin{pmatrix} d_{00} & d_{01} & d_{02} & 0 \\ d_{10} & d_{11} & d_{12} & 0 \\ d_{20} & d_{21} & d_{22} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{8}$$

Y' can be written as $Y + N$, with $N = \begin{pmatrix} n_{00} & n_{01} & n_{02} & n_{03} \\ n_{10} & n_{11} & n_{12} & n_{13} \\ n_{20} & n_{21} & n_{22} & n_{23} \\ n_{30} & n_{31} & n_{32} & n_{33} \end{pmatrix}$, and thus:

$$X' - X = C^T Y' C - C^T Y C = C^T (Y + N) C - C^T Y C = C^T N C = D \tag{9}$$

Each $d_{i,j} = 0$ in Equation 8 corresponds to a linear equation, we thus get seven linear equations in $\overrightarrow{n} = (n_{00}, \ldots, n_{33})^T$.

$$P\overrightarrow{n} = \overrightarrow{0} \tag{10}$$

The solutions of this system of linear equations are spanned by a 9 element basis. Any linear combination of these basis vectors can be added to the de-quantized coefficient matrix without changing the border pixels.

The nine possible basis that would preserve the last row and column for the blocks are given below in equation 11. As explained above, any summation of these basis matrices could be computed and be used as solution patterns that will preserve the "prediction pixels".

$$\left\{ \begin{pmatrix} \frac{1}{4} & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 1 \end{pmatrix}, \begin{pmatrix} -\frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \end{pmatrix}, \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}, \begin{pmatrix} -\frac{1}{2} & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \right.$$
$$\left. \begin{pmatrix} -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} \frac{1}{2} & 0 & 0 & 1 \\ \frac{1}{2} & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} -1 & 0 & 1 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \right\} \tag{11}$$

Our watermark embedding requires two solution patterns of the linear system in order to embed a single symbol of the watermarking sequence. The solutions should modify important coefficients (close to the DC coefficient, but not the DC coefficient itself), but modify as few coefficients as possible, which are also most likely to be non-zero. Hence, we have selected patterns with the minimum amount of non-zero entries that contain non-zero entries close to the DC-coefficient. The following two solution patterns have been employed, each containing 4 non-zero coefficients:

$$sol_1 = \begin{bmatrix} 0\,1\,1\,0 \\ 0\,1\,1\,0 \\ 0\,0\,0\,0 \\ 0\,0\,0\,0 \end{bmatrix} \tag{12}$$

$$sol_2 = \begin{bmatrix} 0\,0\,0\,0 \\ 1\,1\,0\,0 \\ 1\,1\,0\,0 \\ 0\,0\,0\,0 \end{bmatrix} \tag{13}$$

We can notice that sol_1 actually corresponds to the matrix summation of the 8^{th} and 9^{th} elements of the basis matrices, whereas sol_2 corresponds to the matrix summation of the 6^{th} and 9^{th} elements of the basis matrices (see the nine basis matrices in equation 11).

We can thus use either solution sol_1 or sol_2 in order to embed a 1 or a 0 within the block:

$$\begin{cases} y_{1(i,j)} = y_{i,j} + \alpha.sol_{1(i,j)} \rightarrow \text{Embed watermark bit '0'} \\ y_{2(i,j)} = y_{i,j} + \alpha.sol_{2(i,j)} \rightarrow \text{Embed watermark bit '1'} \end{cases} \tag{14}$$

α is a weighting parameter, for any α value, the coefficient used for the prediction will be unchanged in $y_{1(i,j)}$ and $y_{2(i,j)}$. In the following, we will explain how to use a JND model to optimize the embedding strength.

3.2 Watermark embedding

For a better imperceptibility, we have adapted DCTune [26] to H.264/AVC. DCTune performs in the DCT transform domain. Since DCT transform has

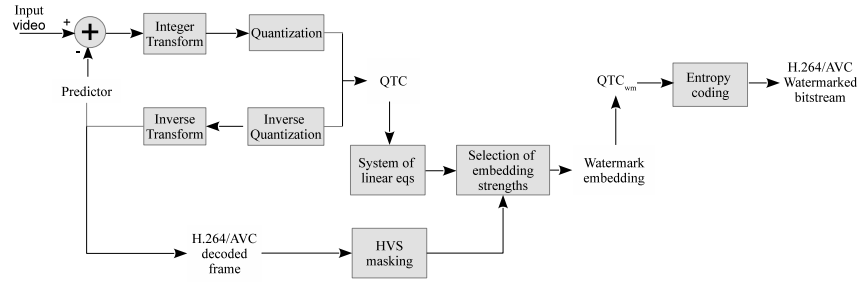
been replaced by integer transform (IT) in H.264/AVC, we have computed the DCTune visibility threshold using the IT. The adaptation of DCTune in our method is performed the same way as in [20]. Interested readers should refer to section II in [20] for in depth details on the DCTune adaptation. In a video frame, every area has different characteristics in terms of luminance, contrast and texture. Hence the sensitivity to human vision also varies from one area to another inside a video frame. DCTune computes a visibility threshold for every transformed coefficient in the DCT domain. DCTune considers both luminance masking and contrast masking. Luminance masking occurs in the bright portions of an image, where basically, our visual system is less sensitive to luminance variation. Effectively, the same luminance variation leads to a lower contrast in brighter areas (than in darker areas). Hence, information loss is less noticeable in bright portions of the image. Concerning contrast masking, the human visual system exhibits a reduced sensitivity when two components having similar frequencies and orientations are mixed together.

Watermark can be embedded in two ways as shown in Fig. 5. If we have raw video input, the watermark embedding and video compression steps can be simultaneously performed as shown in Fig. 5a. A 4x4 block is selected for embedding based on solution of linear equations and DCTune threshold. Since the watermark is embedded in encoded video and not in the original video, DCTune is employed on reconstructed video content as shown in Fig. 5a.

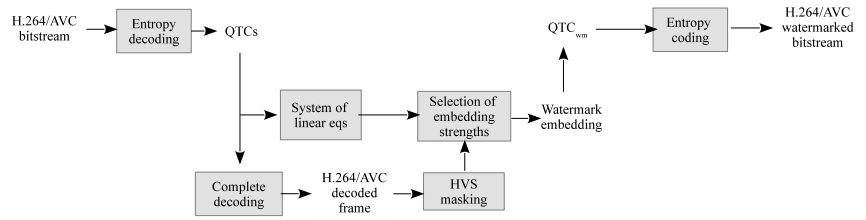
If the input video is in compressed form, we need to perform entropy decoding for watermark embedding as shown in Fig. 5b. In this case, video will be completely decoded for HVS mask creation by DCTune. It is pertinent to mention that complete video decoding is required only once. For example, if

we want to embed 1000 different watermarks in a video bitstream, it would require entropy re-encoding 1000 times but a complete decoding only once.

For both approaches, the solutions of the system of linear equations are employed for watermarking the Quantized Transform Coefficients. One can note that the HVS masking operates on the decoded/reconstructed pixels, which already takes into account the quantization effect.



(a)



(b)

Fig. 5: Block diagram of drift-free robust watermarking system for H.264/AVC, (a) raw video input, and (b) H.264 bitstream input.

As previously explained, for each 4×4 block, a system of 7 linear equations is derived. Evidently, several solutions may exist to this system of equations. In this work, for every "watermarkable" block, we need at least two solutions to exist. Basically, using either of these two solutions, allows us to embed the

watermark information into the host media. In this work, once two solutions to the system are selected, they are used throughout all the blocks of every *intra* frame.

Thus, two solution patterns of the system of linear equations are used to embed a single symbol of a bipolar watermark sequence $W \in \{-1, +1\}$ (with zero mean and unit variance). Consequently, using the two solution patterns, we either encode -1, or +1 within the considered block. During the detection, the cross-correlation of the two solutions with the possibly watermarked block is determined. Thus, the goal is to achieve a maximum difference between these two correlations (cross-correlation gap θ), while keeping the embedding strength within the DCTune visibility threshold. The DCTune JND mask provides a maximum allowable strength for every coefficient within the 4×4 block, thus, the transformed coefficients within the selected embedding patterns (see Equations 12 and 13) are successively increased until their value reaches the DCTune threshold. For every considered 4×4 block, DCTune provides a 4×4 JND matrix. During the perceptual optimization, we use the minimum JND coefficient being non-zero in either sol_1 or sol_2 (see Equations 12 and 13). This minimum perceptual weighting coefficient is then used as α in Equation 14. It is important to note that only the coefficients located within the patterns sol_1 or sol_2 are modified, all other coefficients are kept unchanged. It is important to note that the DCTune mask coefficients are not needed during the detection stage.

In order to determine the required processing power for the proposed algorithm, we performed a simulation on 50 frames of the *tempete* sequence in CIF format with a QP value of 24. For this simulation, three scenarios are considered: 1) Encoding only was performed on the sequence, 2) both encoding

and watermark embedding was performed, and 3) the sequence was encoded and the watermark was weighted using DCTune before the embedding. For this experiment, a 2.1 GHz Intel Core 2 Duo T8100 machine with 3072 MB RAM was used. We have done this simulation both on intra only sequence and on intra & inter sequences. For intra only sequences, where a watermark is embedded within every frame, the encoding itself took 47.9 seconds for the whole sequence of 50 frames. When encoding the sequence and embedding the watermark, it took 48.9 seconds, and finally, when encoding and watermarking were performed using DCTune to weight the watermark strength, 49.3 seconds were needed.

Since intra only sequence is used very rarely and intra, followed by inter frames (IPP sequence) is used most of the time, the simulation was also conducted on intra & inter sequence with intra period of 10. On IP sequences, the encoding itself took 4078 seconds. When encoding the sequence and embedding the watermark, it took 4083 seconds, and finally, when encoding and watermarking were performed using DCTune to weight the watermark strength, 4084 seconds were needed.

The table 1 shows the processing time for all three scenarios. The processing times are given both in terms of average time per frame, and for the whole processing time needed for the 50 frames. We can observe that when I frames only are considered, the processing time is increased by less than 3% compared to the "encoding-only" scenario. When P frames are included in the sequence (intra period=10), the use of DCTune when embedding the watermark only increases the processing time by 0.15 %.

The proposed embedding process is thus slightly faster than the works in [16] where the authors mentioned: *"The data hiding procedure for an I frame of*

Table 1: Processing time when 1) encoding only is applied, 2) encoding and watermarking are applied, and 3) encoding and DCTune adapted watermarking are performed.

		Average processing time (sec / frame)	Global processing time (sec)
I frames	Coding	0.95944	47.972
	Coding&watermark	0.97902	48.951
	Coding & DCTune watermark	0.98668	49.334
I&P frames	Coding	81.56776	4078.388
	Coding&watermark	81.67932	4083.966
	Coding & DCTune watermark	81.69414	4084.707

the sequence Coastguard and Bridge-far cost 188 ms and 125 ms on average, respectively.". It should be noted that in [16], QCIF sequences were used, whereas in Table 1 we used CIF format sequences.

Moreover, in order to ensure a proper quality on Intra + Inter coded sequences, the PSNR was computed on an sequence containing both intra and inter coded frames. The figure 6 shows for every frame of the *tempete* sequence (50 frames) the PSNR when 1) only coding is applied ($QP = 24$) and 2) both coding and watermarking are applied. In this experiment, the intra period is 10, only the I frames are watermarked (frames 1, 11, 21, 31, 41). On average (across the 50 video frames), when encoding only was performed, the PSNR reaches 39.35 dB, and when both coding and watermarking are applied, the PSNR equals 38.89 dB. We can thus witness only a 0.46 dB decrease when the watermarking method is applied.

The following constraints are employed during the watermark embedding:

– Bit-rate constraint: we do not modify zero transform coefficients, i.e., these blocks are not used for embedding.

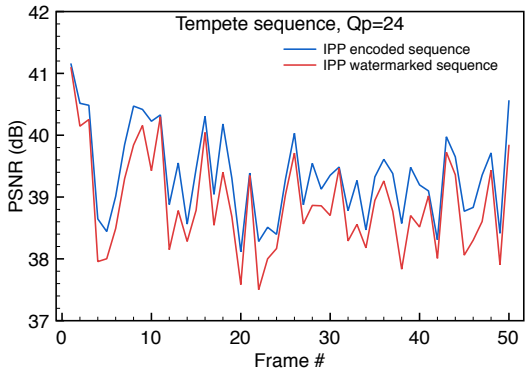


Fig. 6: PSNR computed on every frame of I+P coded sequence for either coding only, or watermarked sequence.

- Imperceptibility constraint: DCTune visibility threshold is calculated for each transformed coefficient. Embedding is performed with the maximum embedding strength allowed by the DCTune visibility threshold.
- Robustness constraint: To guarantee the robustness of this scheme, the cross-correlation gap between the two added solution patterns should be greater than a pre-determined threshold θ.

Embedding can be interpreted as a constrained optimization problem, in which we want to maximize the embedding strength under the constraints of bit rate, imperceptibility and robustness.

When modifying the residuals, several constraints need to be respected:

- A block is considered as watermarkable if at least two solution exist for the system of equations.
- For every watermarkable 4×4 block, the border pixels (x_{03}, x_{13}, x_{23}, x_{30}, x_{31}, x_{32}, x_{33}) must be kept unchanged (see Equation 8).
- Modify the 4×4 blocks having non-zero coefficients at the positions determined in the selected solution patterns (see Eqs. 12 and 13).

– The watermark strength must remain below the visibility threshold as defined by DCTune.
– The cross-correlation gap between the two solution patterns should be above a threshold θ.

3.3 Watermark detection

The detection of the watermark is performed in the compressed domain as well. Its procedure is shown in Figure 7 (within the dotted rectangle). During the watermark detection process, the data required during the retrieval procedure is: 1) the chosen solution patterns of the system of linear equations used for watermarking (sol_1, sol_2) and 2) the watermarked block positions. The proposed detection algorithm is thus semi-blind as some side information needs to be transmitted to the detector. More precisely, 32 bits are needed to encode the solution patterns, and 21 bits are needed for every embedded watermark symbol to specify its particular location within the frame, and within the embedding block.

The detection actually proceeds in two separate steps, in a first step, a watermark is extracted from the potentially watermarked video sequence, and then, during the second step, a global detection is performed where a correlation is computed between the original watermark and its extracted counterpart. These two steps are detailed thereafter.

Watermarked transform coefficients are extracted from the watermarked block. Let Z be the extracted 4×4 block, which can be extracted directly from the bitstream by partially decoding it as shown in Figure 7. We will extract one bit of watermark from the block Z using a normalized cross-correlation

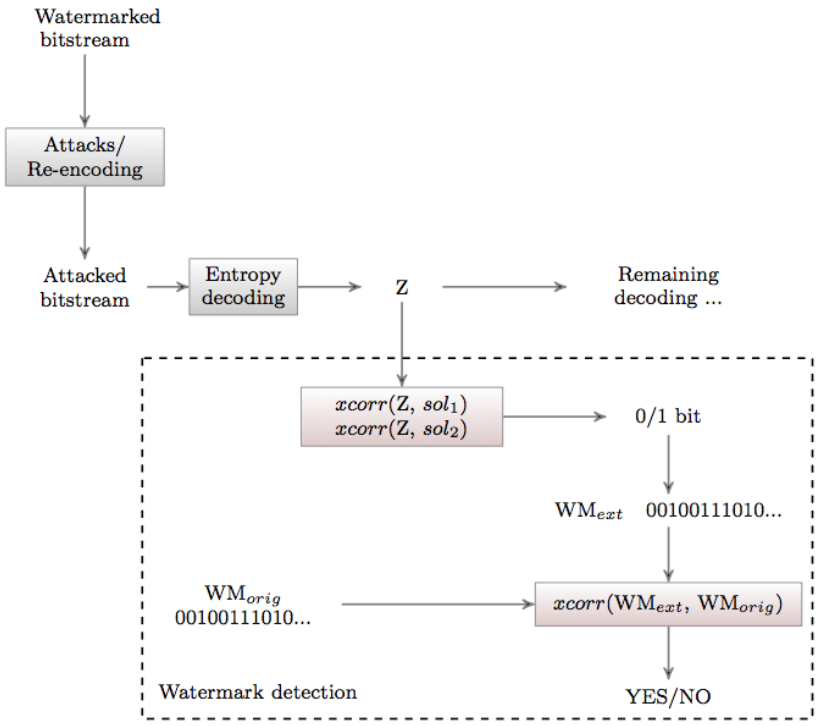


Fig. 7: Watermark detection procedure

with each of (sol_1, sol_2) to find which solution pattern has been embedded as explained in Algorithm 1.

Algorithm 1 Extraction of one bit of watermark message from extracted block Z.

if $xcorr(Z, sol_1) > xcorr(Z, sol_2)$ **then**
  sol_1 detected $\rightarrow$ Extract watermark bit '0'
else
  if $xcorr(Z, sol_2) > xcorr(Z, sol_1)$ **then**
    sol_2 detected $\rightarrow$ Extract watermark bit '1'
  end if
end if
end

Once the extraction of all the watermark symbols is performed, the so obtained watermark sequence WM_{ext} is compared to the original watermark sequence using a normalized cross-correlation. A detection threshold has to be determined, in order to ensure a proper detection [17, 22, 2]. False alarm and missed detection rates were computed in order to select the optimal detection threshold as explained in detail in Section 4. The second cross-correlation differs from the first one in the sense that it is employed to detect the watermark sequence, while the first correlation is employed to extract a symbol of the watermarking sequence, i.e., is employed to determine which solution pattern has been embedded in the block.

4 Simulation Results

This section presents evaluations of the proposed watermarking algorithm [1]. We have used a custom implementation based on the H.264/AVC reference software JM 12.2 [2]. Several benchmark video sequences containing different combinations of motion, texture and objects have been used. In the following, unless otherwise specified, the benchmark sequences were in CIF format (352×288 pixels). In Section 4.1, the choice of a proper detection threshold is presented. The payload and invisibility of our algorithm are assessed as well. Finally, the robustness of the technique is tested against recompression.

[1] A demo is available for download at the following URL: http://www.irccyn.ec-nantes.fr/~autrusse/DEMO/

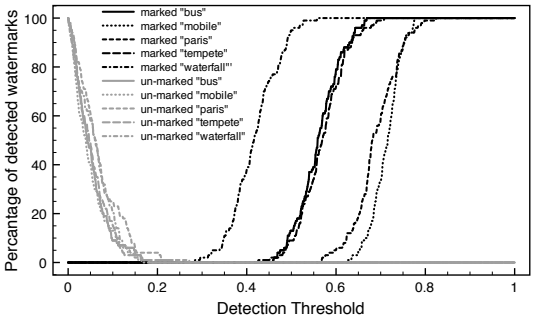
[2] http://iphome.hhi.de/suehring/tml/

4.1 Detection Threshold

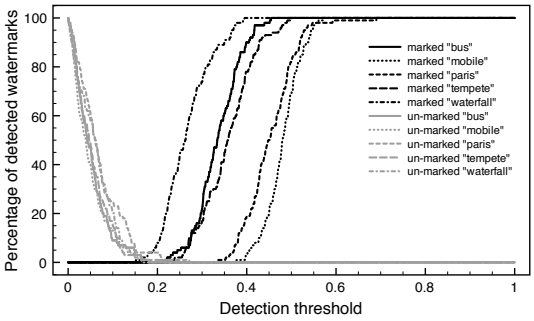
The choice of the optimal detection threshold was obtained by computing the false alarm and missed detection rate on 100 samples, each sample is made of three consecutive *intra* frames. The false positive and false negative rates were computed and led to an optimum detection threshold at 0.3 as depicted in Figure 8. These thresholds allow minimization of false positives, while granting a very weak false negative rate. We deliberately chose here to avoid false positives, in order to ensure that no watermark is detected in an unmarked video. Figure 8 shows the detection performances for five benchmark video sequences. Gray curves indicate the detector result when it is looking for a wrong watermark, while the black curve indicates the results for an original watermark.

In Figure 8(a) the watermarked sequences were re-encoded using a QP value of 34, whereas, in Figure 8(b) the re-encoding used a QP value of 40, and thus the true positives decreases.

Concerning the selection of a detection threshold, two detection scenarios need to be considered. The hypothesis $\mathcal{H}_0$ when no watermark is actually embedded into the host media (or the detector seek for a different watermark), and the hypothesis $\mathcal{H}_1$ when the correct watermark is indeed embedded into the input sequence. The detector is run on a large number of sequences under the two assumptions, $\mathcal{H}_0$ and $\mathcal{H}_1$, and the distribution of the detector is plotted for both cases. This way, two distinct distributions (commonly Gaussian) can be observed while varying the detection threshold. Ideally, the two distributions $\mathcal{H}_0$ and $\mathcal{H}_1$ should not overlap, which then allows us to set a detection threshold in between the two distributions. Distributions for the two hypothe-



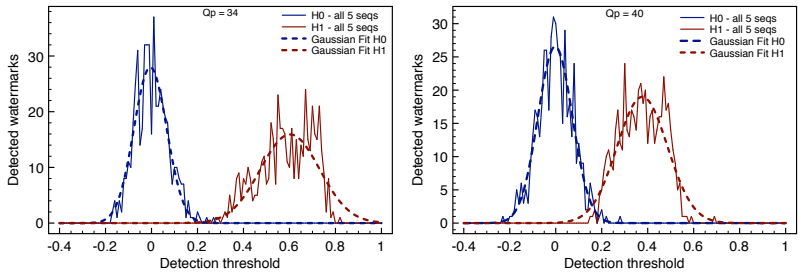
(a) QP = 34



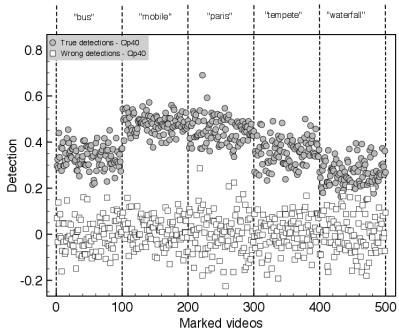
(b) QP = 40

Fig. 8: Experimental analysis for threshold selection (false positives and true negatives).

ses $\mathcal{H}_0$ and $\mathcal{H}_1$ for QP value of 34 and 40 are depicted in Figure 9(a) and Figure 9(b) respectively. Moreover, Figure 9(c) shows all the 500 detections (whether detected or undetected) for all the sequences. In this plot, five input sequences were considered ("bus", "mobile", "paris", "tempete" and "waterfall"), each one was either watermarked with the correct embedding sequence that is used by the detector ($\mathcal{H}_1$, represented by gray circles) or watermarked with a different watermark sequence ($\mathcal{H}_0$, represented by white squares). For each of the five input sequences, 100 watermarked versions were generated, and thus, the $\mathcal{H}_0$ and $\mathcal{H}_1$ distributions are each composed of 500 data points.



(a) QP = 34 (b) QP = 40



(c) Detections for all sequences

Fig. 9: a) $\mathcal{H}_0$ vs. $\mathcal{H}_1$ analysis and detection for all five input sequences.

4.2 Analysis of payload, bit-rate and visual quality

The watermark embedding payload is dependent on two parameters: video content and QP value. The more the video content exhibits textures and motion, the higher is the payload. Table 2 shows the payload, bit-rate increment and PSNR decrease for the benchmark sequences at QP value of 24. The negative value of bit-rate modification actually represents a reduction in bit-rate. Bit-rate change is so small that it is shown in $milli(10^{-3})$ of percent. One can observe that the proposed technique has a negligible effect on PSNR as well, with only 0.064 dB increase, while having an acceptable payload, and a similar bit-rate. It is important to notice that there is a negligible change in the bit-rate despite the fact that we have embedded the watermark message with

the maximum embedding strength allowed by DCTune, to make our scheme more robust.

Table 2: Analysis of the proposed algorithm in terms of payload (per frame), bit-rate and PSNR at QP Values of 24 for benchmark sequences.

Seq	Payload (per frame)	Bit-rate increase (x 10^{-3}%)	PSNR decrease (dB)
"bus"	210.76	-0.52	0.06
"mobile"	275.28	-0.47	0.13
"paris"	117.88	-0.38	0.06
"tempete"	179.24	-0.13	0.05
"waterfall"	173.6	-0.12	0.02
Average	**191.35**	**-0.324**	**0.064**

Table 3: Analysis of the proposed algorithm in terms of payload (per frame), bit-rate and PSNR for different QP Values for the *tempete* sequence.

QP	Payload (per frame)	Bit-rate increase (x 10^{-3}%)	PSNR decrease (dB)
16	250.38	-0.11	0.28
20	221.28	-0.16	0.12
24	179.24	-0.13	0.05
28	132.8	-0.39	0.02
32	72.54	0.12	0
36	23.02	-0.80	0

Video content can be encoded at different quality levels using different QP values. Hence, it is important to observe the payload of the proposed scheme on different QP values. Table 3 shows the analysis of our proposed scheme at different QP values for benchmark video sequence *tempete*. It is evident that the proposed scheme conserves bit-rate and PSNR for the whole range of QP values. Nevertheless, payload reduces with increase in QP value. It is because we embed our watermark symbols only in blocks where solely non-

zero coefficients are affected. The number of such blocks is reduced with the increase in the QP value.

4.3 Robustness against re-compression.

Re-compression is the most traditional non-intentional attack against watermarked videos. Evidently, the higher is the quantization step, the more devastating is the attack. Table 4 summarizes the cross-correlation peaks for five benchmark video sequences for re-compression with 10 different QP values (from 24 to 42 with a step of 2). These results show that the watermark is successfully detected up to a QP value of 40. In Table 4 the gray shaded cells represent the missed detections (the peak correlation is below the detection threshold).

Table 4: Watermark detection rate for benchmark video sequences under re-compression at different QP values (originally encoded at QP value of 24).

Video/QP	24	26	28	30	32	34	36	38	40	42
"bus"	1.00	0.79	0.77	0.74	0.71	0.66	0.54	0.39	0.30	0.23
"mobile"	0.99	0.82	0.81	0.81	0.80	0.71	0.66	0.52	0.56	0.43
"paris"	0.98	0.94	0.84	0.80	0.75	0.69	0.55	0.46	0.33	0.32
"tempete"	1.00	0.83	0.72	0.71	0.67	0.62	0.58	0.52	0.42	0.32
"waterfall"	0.97	0.82	0.78	0.72	0.60	0.51	0.45	0.39	0.34	0.21

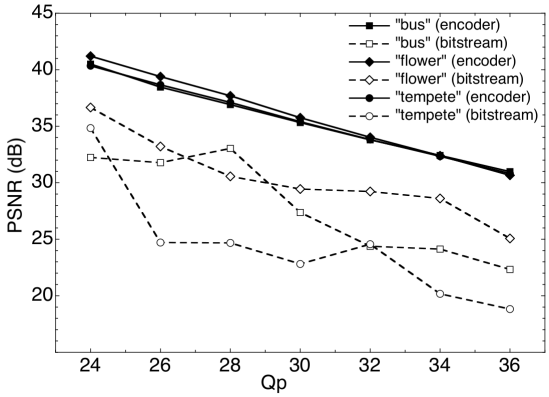
4.4 Comparison with previous works

For the sake of comparison with the recent works on H.264/AVC watermarking, we compared the performances of our watermarking system with [19] and [20] in terms of both PSNR and watermark payload. The PSNR were computed for three watermarked videos, either for the watermarking technique

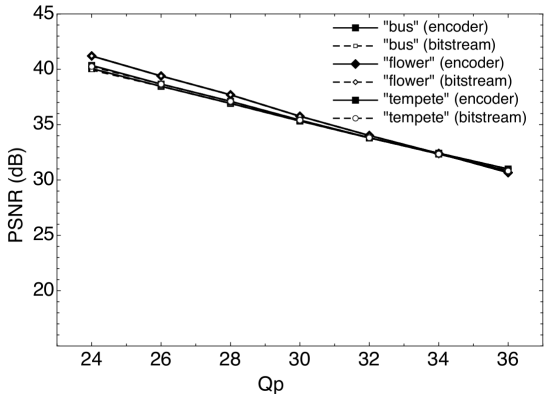
operating within the encoder or in the bitstream. In the following experiments QCIF sequences were used (144×176 pixels), as this was the format used by the authors in [19] and [20]. Figure 10 shows the PSNR for both the proposed technique and [20]. It is important to notice that in [20], the spreading of the prediction error when the watermark is embedded in the bitstream quickly leads to a severe quality loss. On the other hand, PSNR closely overlap in the proposed scheme. For example, for the bus video sequence, the PSNR are respectively 40.05 dB and 39.93 dB in the encoder and in the bitstream. The difference in the PSNR in the proposed scheme is due to the quantization step.

Moreover, in [20], the authors stated: *On average, watermark embedding using our algorithm increases the bit rate of the video by about 5.6% (...) readers might find it useful to know that the PSNR of the watermarked video decreases 0.58 dB compared to the compressed (but unwatermarked) video (...).* As we will see in the following, in the proposed method, the bit-rate increase is marginal, and the PSNR decrease remains very low.

We summarize in Table 5 the performances of our algorithm in terms of bit-rate increase, PSNR decrease, and payload. We highlight the differences in terms of payload between the proposed watermarking system and the works in [19]. We can notice on this table that the two methods present similar watermark payload. Moreover, we also present in Table 5 the payload obtained in [20]. We can notice that the payload from our proposed embedding technique cannot compete with the payload presented in [20], but on the other hand the bit-rate modification is marginal with our method. The authors in [20] did not provide the bit-rate increase per input sequence, but, as explained above, the average bit-rate increase for their method was about 5.6%.



(a) Algorithm [20]



(b) Proposed algorithm

Fig. 10: Comparison of PSNR values with the method presented in [20]. Seven QP values for both the Encoder and Bitstream scenarios.

It is important to notice that the watermarking methods in [19,20] are blind, whereas our algorithm is semi-blind, as previously explained.

Finally, we show in Table 6 a performance comparison with [16] in terms of both bit-rate increase and Payload.

Again, we can witness that the method from [16] offers a higher embedding capacity, but also leads to a higher bit-rate increase. Such an increase could become an important issue when dealing with high definition video sequences.

Table 5: Comparison of payload per frame with the algorithm presented in [19] and [20]. BRi: bit-rate increase, PSNRd: PSNR decrease, P: Payload (in bits per frame)

	Proposed method			from [19]		from [20]
	PSNRd (dB)	BRi (%)	P (bpf)	BRi (%)	P (bpf)	P (bpf)
"Carphone"	0.11	-2.69e-5	37	0.80	44	891
"Claire"	0.14	1.75e-3	34	0.44	22	450
"Mobile"	0.38	-3.51e-3	71	0.23	85	2291
"Mother"	0.14	8.11e-4	36	0.69	42	630
"Salesman"	0.12	1.23e-3	52	N/A	N/A	953
"Soccer"	0.11	-2.14e-5	52	N/A	N/A	N/A
"Table"	0.08	1.09e-3	44	0.31	38	810
"Tempete"	0.14	2.79e-4	45	0.44	81	N/A

Table 6: Comparison with [16] in terms of bit-rate modification and Payload.

	Proposed		Ma et al. [16]	
	BRi (%)	P (bpf)	BRi (%)	P (bpf)
"carphone"	-2.69e-5	37	3.24	806
"mobile"	-3.51e-3	71	0.88	910
"salesman"	1.23e-3	52	7.17	936

5 Discussion

In these experiments, we have only embedded in very specific 4×4 blocks: 1) all of the coefficients involved in watermark embedding are non-zero, 2) the cross-correlation gap between the two watermarking patterns must be greater than θ, and 3) the embedding strength must be within DCTune threshold for that block. If all the three constraints are satisfied, we embed the watermark message bit using the maximum allowable strength by DCTune. In our experiments, we have selected only two solutions to embed a watermark and if any of the above-mentioned constraints are not met for those two solutions, we do not embed any watermark bit in that block.

Our main goal in this work was to provide a robust embedding technique while granting a minimum bit rate modification and high quality marked sequences. We did not intend to provide a high payload embedding. However, it is important to note that the watermark payload, and consequently, the robustness could significantly be improved if more than two solutions were selected. The above-mentioned objectives (robustness, invisibility, bit-rate) were reached using a bipolar watermark. Nevertheless, future works will be devoted to study the possible improvements of using a *multipolar* watermark.

In [20], the authors used a human visual system model adapted for 4×4 DCT blocks, while we have used DCTune for this purpose. Since the embedding position inside the 4×4 block depends on the solutions of the linear system and such solution patterns could be randomly selected to increase security.

We have used the experimental threshold for embedding and cross-correlation to detect the watermark. In [20], the authors establish a theoretical framework for watermark detection, however, this framework only applies for blind detection. Our experiment prove that our algorithm can resist recompression even associated with a high quality degradation.

6 Conclusion

In this paper, we proposed an intra-drift-free robust watermarking algorithm for H.264/AVC in the compressed domain. A system of linear equations is solved to determine additive watermarking patterns for the H.264 4×4 transform coefficients that do not cause intra-drift in the pixel domain. The watermarking strength is adjusted with an adapted DCTune JND mask. A cross-correlation based detection system is employed to guarantee the robustness.

The proposed scheme preserves the bit-rate on average, there is often even a decrease in bit-rate. This is a significant advantage compared to previously proposed approaches, which lead to an increase in bit-rate of the watermarked video. Another advantage of our approach is its runtime-efficiency. Hence the proposed scheme is very suitable for real-time applications, such as video streaming of actively fingerprinted content. At the same time, we see a negligible effect on visual quality (results are given in terms of PSNR). The watermark remains detectable in re-compressed videos, even in very low quality and highly distorted video sequences (encoded with a high QP).

References

1. A. M. Alattar and E. T. Lin and M. U. Celik, Digital Watermarking of Low Bit-Rate Advanced Simple Profile MPEG-4 Compressed Video, IEEE Transactions on Circuits and Systems for Video Technology, 13 (8), 787-800, 2003
2. F. Autrusseau and P. Le Callet, A Robust Image Watermarking Technique Based on Quantization Noise Visibility Threshold, Signal Processing, 87 (6), 1367-1383, 2007
3. C. Chen and J. Ni and J. Huang, Temporal Statistic Based Video Watermarking Scheme Robust against Geometric Attacks and Frame Dropping, Proc. International Workshop on Digital Watermarking, 81–95, 2009,
4. W. Chen and F. Autrusseau and P. Le Callet, A Error-Propagation-Free Perceptual Watermarking Algorithm for H.264/AVC Encoded Video, Proc. 1^{st} Workshop on Visual Signal Processing and Analysis, 2008
5. I. Cox and J. Kilian and F. Leighton and T. Shamoon, Secure Spread Spectrum Watermarking for Multimedia, IEEE Transactions on Image Processing, 6 (12), 1673-1687, 1997
6. F. Deguillaume and G. Csurka and J. O'Ruanaidh and T. Pun, Robust 3D DFT Video Watermarking, proc. SPIE: Security and Watermarking of Multimedia Contents, 3657, 113-124, 1999

7. A. Golikeri and P. Nasiopoulos and Z. Wang, Robust Digital Video Watermarking Scheme for H.264 Advanced Video Coding Standard, Journal of Electronic Imaging, 16(4), 14, 2007

8. X. Gong and H. Lu, Towards Fast and Robust Watermarking Scheme for H.264 Video, Proc. IEEE International Symposium on Multimedia, 649-653, 2008

9. F. Hartung and B. Girod, Watermarking of Uncompressed and Compressed Video , Signal Processing, 66 (3), 283-301, 1998

10. W. Huo and Y. Zhu and H. Chen, A Controllable Error-Drift Elimination Scheme for Watermarking Algorithm in H.264/AVC Stream, IEEE Signal Processing Letters, 18 (9), 535 -538, 2011

11. X. Kang and J. Huang and Y. Shi and Y. Lin, A DWT-DFT Composite Watermarking Scheme Robust to Both Affine Transform and JPEG Compression, IEEE Transactions on Circuits and Systems for Video Technology, 13 (8), 776-786, 2003

12. S. Kapotas and E. Varsaki and A. Skodras, Data Hiding in H.264 Encoded Video Sequences, Proc. IEEE International Workshop on Multimedia Signal Processing, 373-376, 2007

13. S. M. Kim and S. B. Kim and Y. Hong and C. Won, Data Hiding on H.264/AVC Compressed Video, Proc. International Conference on Image Analysis and Recognition, 698–707, 2007

14. G. Langelaar and R. Langendijk, Optimal Differential Energy Watermarking of DCT Encoded Images and Video, IEEE Transactions on Image Processing, 10 (1), 148-158, 2011

15. H. Ling and Z. Lu and F. Zou, New Real-Time Watermarking Algorithm for Compressed Video in VLC Domain, Proc. IEEE International Conference on Image Processing, 4, 2171-2174, 2004

16. X. Ma and Z. Li and H. Tu and B. Zhang, A Data Hiding Algorithm for H.264/AVC Video Streams Without Intra-Frame Distortion Drift, IEEE Transactions on Circuits and Systems for Video Technology, 20 (10), 1320 -1330, 2010

17. M.L. Miller and J.A. Bloom, Computing the Probability of False Watermark Detection, Proc. 3^{rd} International Workshop on Information Hiding, 146-158, 1999

18. B. Mobasseri and Y. Raikar, Authentication of H.264 Streams by Direct Watermarking of CAVLC Blocks, Proc. SPIE: Security, Steganography, and Watermarking of Multimedia Contents IX, 6505, 5, 2007

19. M. Noorkami and R. M. Mersereau, Compressed-Domain Video Watermarking for H.264, Proc. IEEE International Conference on Image Processing, 2, 890-893, 2005

20. M. Noorkami and R. M. Mersereau, A Framework for Robust Watermarking of H.264-Encoded Video With Controllable Detection Performance, IEEE Transactions on Information Forensics and Security, 2 (12), 14-23, 2007

21. M. Noorkami and R. M. Mersereau, Digital Video Watermarking in P-Frames With Controlled Video Bit-Rate Increase, IEEE Transactions on Information Forensics and Security, 3 (3), 441–455, 2008

22. A. Piva and M. Barni and F. Bartolini and V. Cappellini, Threshold Selection for Correlation-Based Watermark Detection, Proc. COST254 Workshop on Intelligent Communications, 67-72, 1998

23. D. Pröfrock and M. Schlauweg and E. Müller, A New Uncompressed-Domain Video Watermarking Approach Robust to H.264/AVC Compression, Proc. IASTED International Conference on Signal Processing, Pattern Recognition, and Applications, 99–104, 2006

24. G. Qiu and P. Marziliano and A. T. Ho and D. He and Q. Sun, A Hybrid Watermarking Scheme for H.264/AVC Video, Proc. 17^{th} International Conference on Pattern Recognition, 4, 865-868, 2004

25. Z. Shahid and M. Chaumont and W. Puech, Considering the Reconstruction Loop for Data Hiding of Intra and Inter Frames of H.264/AVC, Signal, Image and Video Processing, Springer, 5, 2, 2011.

26. A. Watson, DCT Quantization Matrices Visually Optimized for Individual Images, Proc. SPIE Society of Photo-Optical Instrumentation Engineers, 1913, 202-216, 1993

27. X. Wu and W. Zhu and Z. Xiong and Y. Zhang, Object-Based Multiresolution Watermarking of Images and Video, Proc. IEEE International Symposium on Circuits and Systems, 1, 212-215, 2000

28. D. Zou and J. Bloom, H.264/AVC Substitution Watermarking: A CAVLC Example, Proc. SPIE, Media Forensics and Security XI, 7254-29, 2009

29. D. Zou and J. Bloom, H.264 Stream Replacement Watermarking with CABAC Encoding, Proc. IEEE International Conference on Multimedia and Expo, 117-121, 2010

30. G. Wu and Y. Wang and W. Hsu, Robust Watermark Embedding/Detection Algorithm for H.264 video, Journal of Electronic Imaging, 14(1) , 13013, 2005

31. J. Oostveen and T. Kalker and J. Haitsma, Visual Hashing of Digital Video: Applications and Techniques, Proc. SPIE, Applications of Digital Image Processing XXIV, 4472, 121-131, 2001

32. L. Qiao and K. Nahrstedt, Non-invertible watermarking methods for MPEG encoded audio. In SPIE Security and Watermarking of Multimedia Data, 3657, 194-203, 1999