J Real-Time Image Proc (2009) 4:33–41
DOI 10.1007/s11554-008-0100-2

SPECIAL ISSUE

Real time data hiding by exploiting the IPCM macroblocks in H.264/AVC streams

Spyridon K. Kapotas · Athanassios N. Skodras

Received: 15 June 2008 / Accepted: 19 September 2008 / Published online: 11 October 2008

Abstract A new method for data hiding in H.264/AVC streams is presented. The proposed method exploits the IPCM encoded macroblocks during the intra prediction stage in order to hide the desired data. It is a blind data hiding scheme, i.e. the message can be extracted directly from the encoded stream without the need of the original host video. Moreover, the method exhibits the useful property of reusing the compressed stream for hiding different data numerous times without considerably affecting either the bit-rate or the perceptual quality. This property allows data hiding directly in the compressed stream in real time. The method perfectly suits to covert communication and content authentication applications.

Keywords H.264/AVC · Intra prediction · IPCM · Data hiding · Video authentication · Covert communication

1 Introduction

The widespread use of the Internet and World Wide Web has changed the way digital data is handled. The easy access of images, musical documents and movies has modified the development of data hiding, by placing emphasis on copyright protection, content-based authentication, tamper proofing, annotation and covert communication. Data hiding deals with the ability of embedding data into a digital cover with a minimum amount of perceivable degradation, i.e., the embedded data is invisible or inaudible to a human observer. Data hiding consists of two sets of data, namely the *cover medium* and the *embedding data*, which is called the *message*. The cover medium and the message can be text, audio, picture or video depending on the size of the message and the capacity of the cover.

Data hiding and watermarking techniques are usually studied together because a watermarking technique can serve as a data hiding technique, as well, although the opposite is not always feasible. Early video data hiding approaches were essentially still image watermarking techniques extended to video by hiding the message in each frame independently [1]. Methods such as spread spectrum were used, where the basic idea is to distribute the message over a wide range of frequencies of the host data. Transform domain is generally preferred for hiding data since, for the same robustness as for the spatial domain, the result is more pleasant to the Human Visual System (HVS). For this purpose the Discrete Fourier Transform (DFT), the Discrete Cosine Transform (DCT), and the Discrete Wavelet Transform (DWT) domains were usually employed [2–4]. Recent video data hiding techniques are focused on the characteristics generated by video compressing standards. Motion vector based schemes have been proposed for MPEG algorithms [5–7]. Motion vectors are calculated by the video encoder in order to remove the temporal redundancies between frames. In these methods the original motion vector is replaced by another locally optimal motion vector to embed data. Only few data hiding algorithms considering the properties of H.264 standard [8–13] have recently appeared in the open literature. In [8] a subset of the 4×4 DCT coefficients are modified in order to achieve a robust watermarking algorithm for

S. K. Kapotas (✉) · A. N. Skodras
Digital Systems and Media Computing Laboratory,
School of Science and Technology,
Hellenic Open University, Patras, Greece
e-mail: s.kapotas@eap.gr

A. N. Skodras
e-mail: skodras@eap.gr

H.264. In [9] the blind algorithm for copyright protection is based on the intra prediction mode of the H.264 video coding standard. In [10] some of the skipped macroblocks are used to embed data. The algorithm in [11] hides one bit in each qualified intra 4 × 4 luma block by modifying the intra 4 × 4 prediction modes. In [12] data hiding is based on the block size selection during the inter prediction process of the H.264 encoding. In [13] one bit of information is embedded in the sign bit of the trailing ones in Context Adaptive Variable Length Coding (CAVLC) of the H.264/AVC stream.

Traditionally, most of the data hiding techniques take place during the encoding either in the spatial or in the transform domain. Thus, for these techniques it is extremely difficult to perform data hiding in real time. Moreover, the access to the original video sequence is not always feasible and reuse of the marked video to hide different data is necessary. First decoding of the bitstream and then re-encoding it to embed the new data is needed. However, this results in video quality degradation, since the data hiding process has always some negative impact to the PSNR. Moreover, reusing the marked video, by decoding and re-encoding it, cannot always take place in real time. Other techniques, applied in the compressed domain [5, 8, 11], may produce artifacts and drift errors, which degrade the video quality. Such errors make the reusing of the marked video in the compressed domain very ineffective. The proposed method addresses the aforementioned problems, i.e. data hiding in real time and reusing the marked video in real time without affecting video quality.

In general, the well-established H.264/AVC video coding standard [14] has adopted many advanced features, which can be exploited for data hiding purposes. In this paper we propose a new data hiding scheme, which exploits the IPCM mode used by the H.264 encoder during the intra prediction, in order to hide the desired data. The data can then be extracted directly from the encoded stream without using the original host video. The whole process is depicted in Fig. 1. This method is best suited to content-based authentication and covert communication applications.

The rest of the paper is organized as follows. In Sect. 2 we briefly present the Intra Prediction within the H.264/AVC encoder [15]. In Sect. 3 we describe the new method and in Sect. 4 the message extractor. Simulation results are presented in Sect. 5. Some considerations for further improvements are given in Sect. 6. Finally, in Sect. 7 conclusions are drawn.

2 Intra mode prediction in H.264/AVC

The H.264/AVC standard supports coded sequences containing I and P slices. I slices contain intra coded

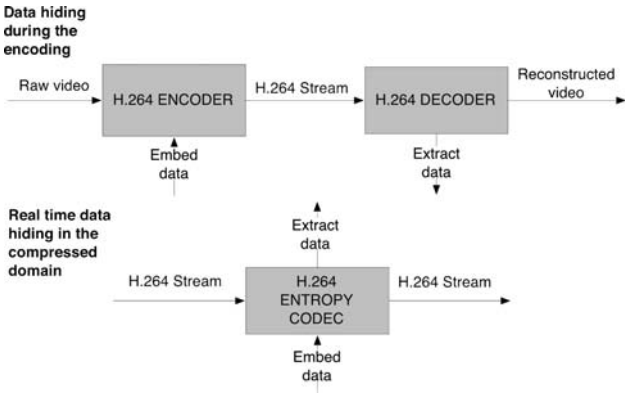

Fig. 1 The process of embedding and extracting data using the H.264 codec

macroblocks in which each 16 × 16 (I_16 × 16) or 4 × 4 (I_4 × 4) luma region and each 8 × 8 (I_8 × 8) chroma region is predicted from previously-coded samples in the same slice. A third type of Intra coding, called IPCM, is also provided for use in unusual situations. The encoder typically selects the prediction mode for each block that minimizes the difference between the predicted block and the block to be encoded.

The I_4 × 4 mode is based on predicting each 4 × 4 luma block separately and is well suited for coding parts of a picture with significant detail. The I_16 × 16 mode, on the other hand, performs prediction and residual coding on the entire 16 × 16 luma block and is more suited for coding very smooth areas of a picture. In addition to these two types of luma prediction, a separate chroma prediction is conducted. In contrast to previous video coding standards (especially H.263 + and MPEG-4 Visual), where intra prediction has been conducted in the transform domain, intra prediction in H.264/AVC is always conducted in the spatial domain, by referring to neighboring samples of previously-decoded blocks that are on the left and/or above the block to be predicted. Since this can result in spatio-temporal error propagation when inter prediction has been used for neighboring macroblocks, a constrained intra coding mode can alternatively be selected that allows prediction only from intra-coded neighboring macroblocks. In I_4 × 4 mode, each 4 × 4 luma block is predicted from spatially neighboring samples.

When the fidelity of the coded video is high (i.e., when the quantization step size is very small), it is possible in certain very rare instances of input picture content for the encoding process to actually cause data expansion rather than compression. Furthermore, it is convenient for implementation reasons, to have a reasonably low identifiable limit on the number of bits necessary to process in a decoder in order to decode a single macroblock. To address these issues, the standard includes an IPCM macroblock

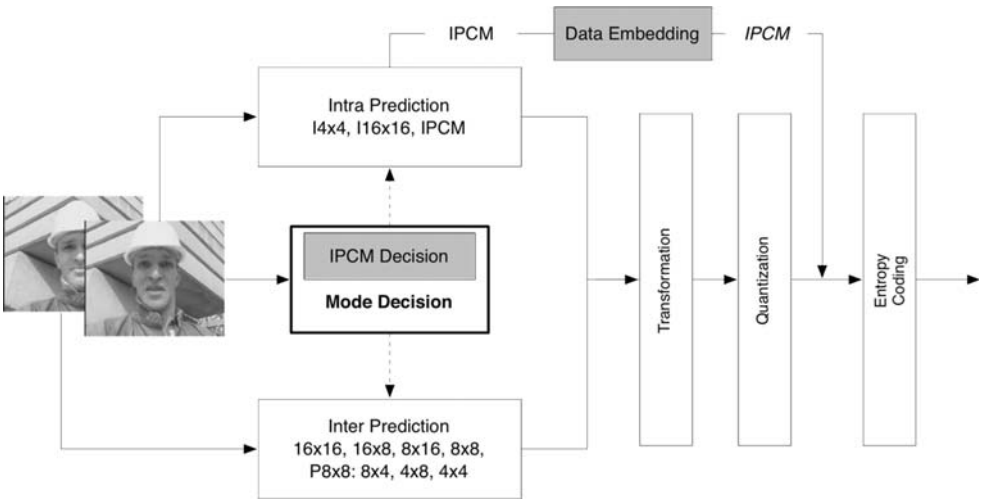

Fig. 2 Simplified block diagram of the proposed method integrated within the H.264 encoder

mode, in which the values of the samples are sent directly—without prediction, transformation, or quantization. An additional motivation for support of this macroblock mode is to allow regions of the picture to be represented without any loss of fidelity. However, the IPCM mode is clearly not efficient—indeed it is not intended to be efficient—rather, it is intended to be simple and to impose a minimum upper bound on the number of bits that can be used to represent a macroblock with sufficient accuracy. If one considers the bits necessary to indicate which mode has been selected for the macroblock, the use of the IPCM mode actually results in a minor degree of data expansion.

3 The proposed data hiding approach

As mentioned in Sect. 2, 'IPCM' is a macroblock in which the values of the samples are sent directly—without prediction, transformation, or quantization. The concept behind our method is to hide the desired data in the low bits of both the luma and the chroma samples of an IPCM macroblock. Eventually, the hidden data will be embedded into the compressed H.264 stream intact. Simple and straightforward though, the proposed method has to face two practical obstacles: the rareness of the IPCM macroblocks during the encoding and the low efficiency of the IPCM mode in terms of compression. The latter turns out to be a trade off issue between the generated bitrate and the payload of the hidden data. This issue is discussed in Sect. 5 with the help of some experimental results. Regarding the rareness of the IPCM macroblock, we conducted several tests with many well-known video sequences. All of the tests resulted in none IPCM macroblocks no matter how low the quantization parameter was set. We therefore concluded that the only safe way to produce IPCM macroblocks during the encoding is to force the encoder to regard specific macroblocks as IPCM macroblocks. The simplified block diagram of the proposed method integrated within the H.264 encoder is shown in Fig. 2.

According to the proposed method, two new steps are added in the H.264 encoder: the *IPCM Decision* and the *Data Embedding*.

- *IPCM Decision* intervenes in the Mode Decision process of the H.264 encoding and forces certain macroblocks to be encoded as IPCM macroblocks. The decision, on which macroblocks are going to be encoded as IPCM, depends on the length of the data to be hidden. The rule is that the IPCM macroblocks must be enough to cover the hidden data and must be spread to the possible extend within the H.264 stream. In the rare situation that the encoder decides to encode a macroblock as IPCM, without our intervention, the *Data Embedding* step will also use this macroblock to hide data.
- *Data Embedding* takes action after the *IPCM Decision* and modifies the low bits of the values of the aforementioned macroblocks in such a way that the modified bits form the hidden data. The "tweaked" *IPCM* macroblocks will then undergo the lossless entropy encoding and the hidden data will eventually be inserted intact into the generated H.264 stream.

The proposed method is characterized by three main features, namely, *ease of implementation*, *respectable data capacity*, and *reusability*. The latter allows data hiding in real time directly in the compressed domain. All these features are described below.

3.1 Ease of implementation

The proposed method can be easily integrated within the reference H.264 encoder [15]. It takes place in a very early stage of the encoding process, before any spatial or

temporal predictions and before the transformation and the quantization. Therefore, the impact of the proposed method to the encoding process is minimized. Some implementation hints on how to implement the proposed algorithm using the reference H.264 encoder version JM14.0 [15] are given below:

1. Add the following code just before the "*compute_mode_RD_cost*" function is called:
   if(img->current_mb_nr==N && img->type==P_SLICE)
   for (i = 0; i < 11; i++) enc_mb.valid[i] = 0;
   This code will force the encoder to encode the Nth macroblock of every P slice in IPCM mode.
2. Modify the low bits of the values of this macroblock. This is done under the "*IPCM*" case inside the "*RDCost_for_macroblocks*" function.

The two steps above correspond to the *IPCM Decision* and the *Data Embedding* blocks, respectively.

3.2 Data capacity

The data capacity of a video sequence (YUV 4:2:0) is calculated in accordance with Eq. 1:

$$\text{Data capacity} = \text{Luma capacity} + 2 \times \text{Chroma capacity} \quad (1)$$

where

$$\text{Luma capacity} = 256 \times N_{\text{IPCM}} \times \text{L}_{\text{bits}}, \text{ and}$$
$$\text{Chroma capacity} = 64 \times N_{\text{IPCM}} \times \text{C}_{\text{bits}}$$

N_{IPCM} is the number of the IPCM macroblocks used for data hiding, L_{bits} is the number of the low bits per IPCM luma sample used for data hiding, and C_{bits} is the number of the low bits per IPCM chroma sample used for data hiding. The luma and chroma samples are 256 and 64, respectively.

According to [16], up to three low bits of an 8-bit sample can be modified without causing any visual distortion. However, our experiments showed that even if the four low bits are modified the distortion is imperceptible. This is explained by the fact that we are not dealing with static images but with moving frames at a rate of 30 fps. Moreover, we embed no more than one IPCM macroblock per frame and not in successive frames. Finally, the rest of the non-IPCM macroblocks have possibly suffered greater distortion due to the intra/inter prediction and to the quantization during the encoding. In order to prove the above we zeroed the four low bits of every luma and chroma sample of the 49th macroblock of the 9th frame of the *mobile* sequence. The sequence was encoded (QP = 28, CABAC) and decoded using the H.264 JM.14.0 codec. Figure 3 shows the visual result.

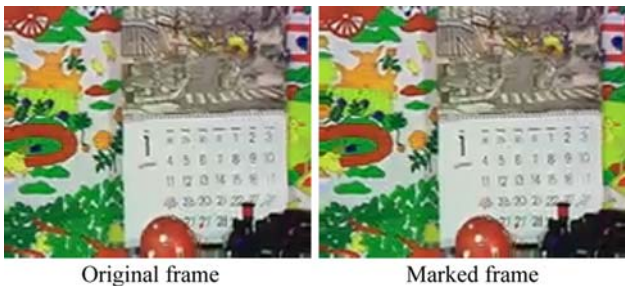

Fig. 3 Comparison of the visual results between the original and the marked 9th frame of *mobile*

An interesting approach would be the modifiable bits L_{bits} and $\text{C}_{\text{bits}}$to be mathematically related to the Quantization Parameter (QP). For example for a high QP (>28) we could modify four bits while for a lower QP we could modify three bits or fewer. Other combinations are also applicable such as the use of three bits for the luma blocks and four bits for the chroma blocks. In the current implementation of the proposed method we modify the four low bits of both of the IPCM luma and the chroma samples in order to hide the data. Hence, from Eq. 1, a single IPCM macroblock ($N_{\text{IPCM}} = 1$) for $\text{L}_{\text{bits}} = \text{C}_{\text{bits}} = 4$ gives a capacity of 1,536 bits. This might be regarded as the upper limit of the capacity per IPCM macroblock.

3.3 Reusability and real time data hiding

Most of the data hiding methods hide the data during the encoding process, thus they are slow and ineffective for real time applications such as covert mobile communication. The proposed method, as explained above, encodes some macroblocks in IPCM mode and hides the message within their data. After the first pass and as long as the IPCM macroblocks have been encoded, the same IPCM macroblocks can be reused to hide new data, directly in the compressed domain, numerous times in real time. The reusing process needs neither the original video sequence nor the original encoded stream. Furthermore, it is proved that it does not cause any considerable PSNR or bit rate distortions. The IPCM macroblock reuse is described below.

The H.264 bitstream is organized in slices, each containing a number of coded macroblocks. Figure 4 shows the simplified syntax of a coded slice. All of the Headers and the Payloads are entropy encoded. The MB Header contains, among others, the prediction modes of the macroblock. The MB Payload usually contains the coded transform coefficients corresponding to the residual image samples after prediction. However, in case of an IPCM macroblock, the Payload contains the pixel values.

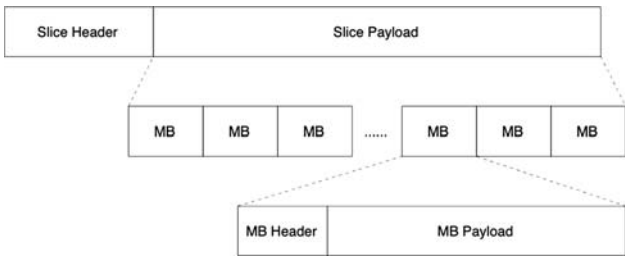

Fig. 4 Simplified syntax of a coded slice

The reusing process is performed in four steps, as follows:

1. Get a slice unit from the H.264 stream.
2. Entropy decode each macroblock's header to determine if this is an IPCM macroblock.
3. In case of an IPCM macroblock:
   a. entropy decode the macroblock payload in order to get the pixel values,
   b. hide the new data into the low bits of the pixels,
   c. entropy re-encode the macroblock,
   d. store the macroblock back to the slice unit.
4. Go to step 1.

The real time data hiding is achieved by the fact that the method needs only to entropy decode and re-encode the compressed IPCM macroblocks, thus avoiding the time-consuming normal encoding process. Figure 5 shows the block diagram of the real time data hiding process.

4 Message extractor

The message extractor is a software tool, not necessarily an H.264 decoder, which extracts the hidden message from the marked H.264 bitstream. The message extraction process is as follows:

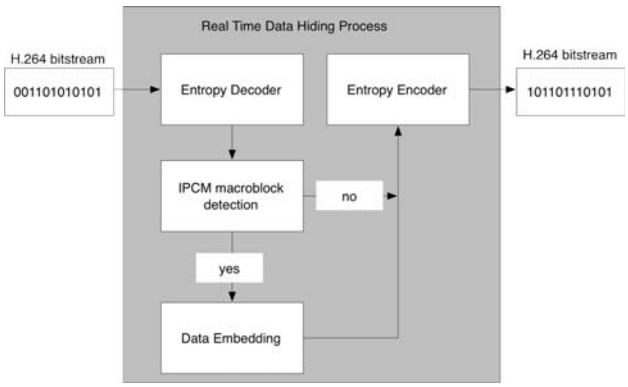

Fig. 5 Block diagram of the real time data hiding process

1. Get a slice unit from the H.264 stream.
2. Entropy decode each macroblock's header to determine if this is an IPCM macroblock.
3. In case of an IPCM macroblock:
   a. entropy decode the macroblock payload in order to get the pixel values,
   b. read the low bits of the pixels in order to extract the message.
4. Go to step 1.

The message extractor works in a way similar to that part of the algorithm that reuses the IPCM macroblocks for data hiding. This gives it the desirable property of being of very low complexity and thus appropriate for real time applications, as it is required in most cases of video [8].

5 Simulation results

The proposed algorithm was integrated within version JM14.0 of the reference H.264 software [15]. The most important configuration parameters of the reference software are given in Table 1. The rest of the parameters have retained their default values.

The IPCM macroblocks are expected to have a negative impact to the produced bit rate. We conducted several tests in order to investigate this impact. For that purpose we used 300 frames or 10 s of well-known representative video sequences in QCIF format (YUV 4:2:0) such as the *akiyo* (Class A), the *foreman* (Class B) and the *mobile* (Class C). The QCIF format (176 × 144) was chosen because it is very common in mobile applications where the demand for real time is always high. The three classes that we used for our experiments possess the following characteristics:

Class A: Low spatial detail and low amount of movement;
Class B: Medium spatial detail and low amount of movement or vice versa;
Class C: High spatial detail and medium amount of movement or vice versa.

Table 1 Configuration parameters of the encoder

Profile	Main
Number of frames	300 (10 s)
Frame rate	30 fps
RD optimization	High complexity mode
Motion estimation	Simplified UMHexagonS
Intra period	0: only the first frame is intra
Symbol mode	CABAC

The hidden message was generated by the pseudorandom integer generator function, *rand*, which is provided by the standard C library. The testing procedure was to run the reference encoder with and without our algorithm and then compare the results with respect to bit rate and PSNR. We used the bit rate and the PSNR variation as comparative metrics, which were calculated as in Eqs. 2 and 3, respectively:

$$\mathrm{VAR}_{\mathrm{RATE}} \frac{R' - R}{R} \times 100(\%) \tag{2}$$

where R' is the bit rate generated by the modified encoder and R is the bit rate generated by the reference encoder, and

$$\mathrm{VAR}_{\mathrm{PSNR}} = \frac{\mathrm{PSNR'}_Y + \mathrm{PSNR'}_U + \mathrm{PSNR'}_V - \mathrm{PSNR}_Y - \mathrm{PSNR}_U - \mathrm{PSNR}_V}{3} \times 100(\%)$$

where PSNR′ is the total PSNR of the luma (Y) and chroma (U,V) samples generated by the modified encoder and PSNR is the total PSNR of the luma and chroma samples generated by the reference encoder.

All of the values of the R', R, PSNR′ and the PSNR were read from the log.dat file, which is created by the encoder's intrinsic logging mechanism. At the first series of tests we ran the encoder for different very common Quantization Parameters (10, 20 and 30) and for different message payloads (3,072–18,432 bits or 10.2–61.4 bits per QCIF frame). Moreover, we did not apply any bit rate constraints. In this way the PSNR remained practically unaffected with the exception of the QP = 10 test case, where the PSNR showed some degradation of less than −0.1 dB. The results are shown in Figs. 6, 7 and 8.

From the results we see that the bit rate is increased proportionally to the data capacity and to the quantization parameter. This is expected because the higher the quantization parameter is, the lower the produced bit rate under the standard reference encoding. On the other hand, our method increases the bit rate proportionally to the IPCM macroblocks, i.e. to the data payload. For QP ≤ 20 the method presents a rather uniform behavior with a bit rate variation from −0.1 to ∼4.0% even for the high payloads. However, for QP ≥ 20 the method results in a continuous bit rate increase. It is notable that *akiyo* sequence presents much higher bit rate variations than the other two sequences. This is due to the fact that *akiyo* is a class "*A*" sequence, i.e. it has low spatial details and low amount of movement. This means that both of the intra and the inter prediction leave small residuals during the normal encoding by the reference encoder, which eventually results in a very low bit rate. On the other hand, our modified encoder always produces the required IPCM macroblocks, which slightly increase the bit rate. Therefore, we conclude that for class "*A*" sequences and for high QPs (>20) the method cannot efficiently hide more than 5,000 bits in 10 s of video, i.e. an average of 16.7 bits per QCIF frame.

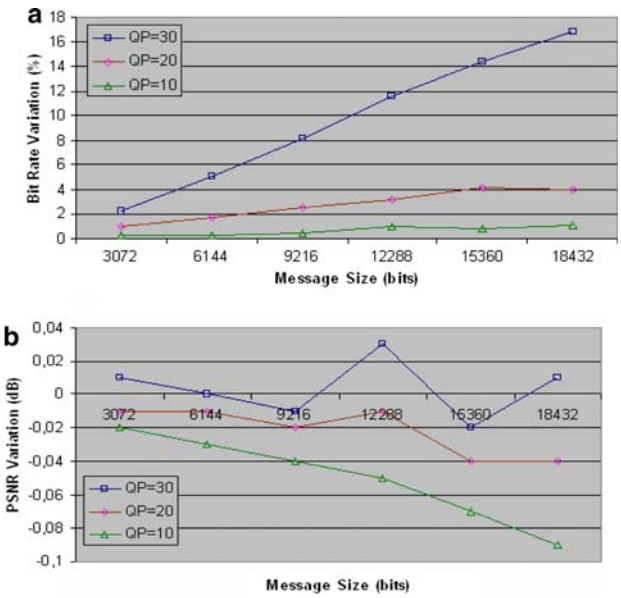

Fig. 6 Bit rate (**a**) and PSNR (**b**) variations versus message size for different QP parameter values for the *Akiyo* sequence (300 frames)

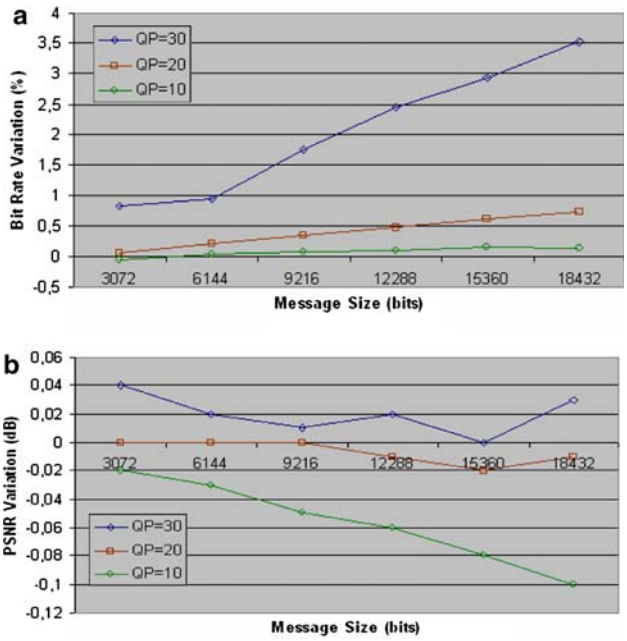

Fig. 7 Bit rate (**a**) and PSNR (**b**) variations versus message size for different QP parameter values for the *Foreman* sequence (300 frames)

In the second series of tests we enabled the bit rate control mechanism of the encoder and we set 60 Kbps and 50 Kbps bit rate constraints on the encoder, which are considered to be low bit rates for the today's Internet

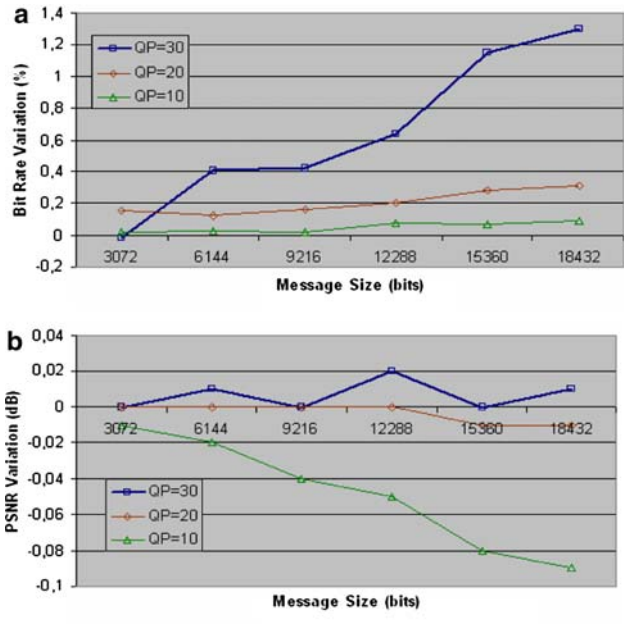

Fig. 8 Bit rate (**a**) and PSNR (**b**) variations versus message size for different QP parameter values for the *Mobile* sequence (300 frames)

Table 2 Bit rate variations under bit rate control

Sequence	Average bit rate variation (%)	
	50 Kbps	60 Kbps
Akiyo	0.05	0.06
Foreman	−0.02	−0.28
Mobile	0.03	−0.04

standards. Our purpose was to investigate the performance of our method when the marked H.264 bitstream has to be transmitted over a channel with limited bandwidth. By enabling the bit rate control the quantization parameters were automatically controlled by the encoder, which had to generate a bit rate lower or equal to the bit rate constraint. In this way the bit rate was practically unaffected as shown in Table 2. Apparently, the cost of doing so was put on the PSNR. Figure 9 shows the PSNR variations when the bit rate constraint is enabled.

From these results it is seen that the overall performance becomes smoother when the bit rate control is enabled. PSNR decreases proportionally to the message payload. The maximum reduction does not exceed 0.43 and 0.44 dB for the 60 and the 50 Kbps constraints, respectively, at a payload of 18,432 bits for the 300 QCIF frames, i.e. an average of 61.4 bits per QCIF frame. These reductions are observed for the *akiyo* sequence, as expected. The result is regarded as an acceptable trade off, taking into account the low bit rate constraints and the small number of frames used (i.e. 300).

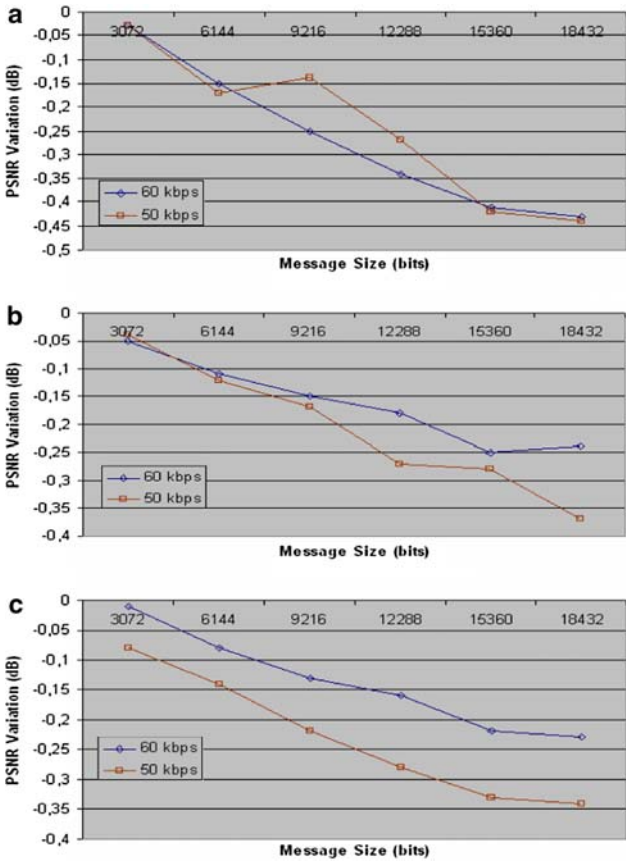

Fig. 9 PSNR variation vs message size for different bit rates for the video sequences (**a**) Akiyo, (**b**) Foreman, and (**c**) Mobile (300 frames)

In the third series of tests we compared our method with the method proposed by Hu et al. [11]. Four QCIF sequences were used (*Bridge Close*, *Grandma*, *News*, and *Silent* of 199 frames each). The tests were performed using H.264 Main Profile configuration (with RDO, CABAC, QP = 28, and 30 frames/s) and a GOP structure of "IBPBPBPBPB". Our method was integrated within the H.264 reference encoder JM14.0 while the competitive method had been integrated in the H.264 reference encoder JM11.0. The results are shown in Table 3. The *PMC* denotes the maximum payload for the proposed method while the *11MC* denotes the maximum payload for method [11] that was used in the comparisons.

Finally, our experiments showed that the proposed algorithm did not introduce serious delays in the encoding process. When the method works in the compressed domain it manages to hide the same amount of data as in the uncompressed domain in real time. This is feasible because the method has to entropy decode a few macroblocks only. Since entropy decoding does not incorporate the inverse transform, it is performed very fast. Moreover, embedding the data by tampering the low bits of the macroblock's pixels is done using logical operations

Table 3 Comparison results between the proposed method and that of Hu et al. [11]

Sequence PMC/11MC	PSNR variation (dB)		Bit rate variation (%)	
	Proposed	[11]	Proposed	[11]
Grandma 15360/12352	−0.01	−0.08	2.93	3.72
Bridge-close 15360/11748	0.01	−0.04	1.15	2.90
News 18432/9972	−0.01	−0.01	3.80	3.23
Silent 18432/17368	0.02	−0.02	3.59	4.14

(AND, OR) and, therefore, it adds a negligible delay to the whole data hiding process. For the above reasons, the proposed method works, in the compressed domain, much faster than the H.264 decoder.

Based on all of the above results, the conclusion is that the proposed method is fast and works better for bit rates around 60 Kbps and higher, where the maximum PSNR degradation does not exceed 0.43 dB for the higher payloads.

6 Further improvements

The proposed approach can be modified in a number of ways in order to enhance its features, namely maximum payload, PSNR, bit rate, and robustness. It is noted that not all of the above can be simultaneously improved.

A potential improvement is in the way the proposed method chooses the IPCM macroblocks. Currently the method chooses the IPCM macroblocks taking into account only the amount of data to be embedded. However, this may result in unjustifiable bit rate expansion. For example the method may choose to force the encoder to encode a specific macroblock as IPCM, while the encoder would have encoded it as a skip one. It would be better for the method to let the encoder perform the motion estimation first. Then, only the macroblocks, which presented non-zero residuals during the motion estimation, would be considered to be candidate IPCM macroblocks. In that way the impact on the bit rate would be smaller. Preliminary simulation tests have shown that there are no considerable variations with the results presented in the previous section. This was expected, as IPCM macroblocks constitute a limited part of the frame. Along the same lines, the message data could be unevenly distributed in the macroblocks, based on each block's statistics (i.e. roughness), thus further reducing the resulted perceptual degradation. This means that a different number of the lower bits of each IPCM macroblock could be devoted to data hiding. Also, instead of choosing a particular macroblock of each slice P to hide data, this could be chosen in a random way based upon a key.

In general, the fact that the proposed method inserts raw information (IPCM macroblocks with modified image data) into the H.264 bitstream generates a lot of potential improvements. The IPCM macroblock can be regarded as part of a still image. Therefore, many data hiding and watermarking techniques, which work in the spatial domain, can be applied [17]. For example, the message could be first encrypted by means of a key, before being embedded into the macroblocks.

7 Discussion and conclusions

In this paper a new low complexity data hiding scheme for H.264 encoded video sequences has been presented. Embedding takes place during the encoding process and exploits the IPCM coded macroblocks in order to hide the data. The same IPCM macroblocks can be reused to hide new data, directly in the compressed domain, numerous times in real time. The method is blind, meaning that the original host video sequence is not needed for message extraction. The method achieves relatively high data capacities without considerably affecting either the video quality or the coding efficiency. The method is fragile, which means that if the marked video is decoded and re-encoded, the embedded data will be lost. The method could be used in a number of applications such as content authentication, tamper proofing, covert communication, advertisement monitoring, content indexing and archiving, etc.

The fact that the proposed method embeds the data directly in the spatial domain (pixel values) makes it immune to some very common errors, which appear in other competitive data hiding techniques working in the compressed domain, such as artifacts and drift. Artifacts are produced by the data hiding techniques, which use the transform coefficients in order to embed the data. Transform coefficient modification results in signal reconstruction errors with undesirable visual effects, such as discontinuities in the block edges. Drift errors are propagating visual distortions between successive video frames as a result of embedding data in the compressed domain without re-performing motion estimation.

In conclusion, the main advantages of the proposed method are its low complexity and the possibility of using the compressed stream for hiding different data many times, without first decoding and then re-encoding the video sequence. This makes the method appropriate for real-time applications. Simulation results have shown that perceptual quality is preserved without sacrificing coding efficiency.

Acknowledgments This work was funded by the European Union, European Social Fund (75%), the Greek State, Ministry of Development, General Secretariat of Research and Technology (25%) and the Private Sector in the frames of the European Competitiveness Program (Third Community Support Framework, Measure 8.3, Program PENED, contract no. 03EΔ832).

References

1. Chae, J.J., Manjunath, B.S.: Data hiding in video. In: IEEE Proceedings of International Conference on Image Processing (ICIP), pp. 243–246 (1999)
2. Fotopoulos, V., Skodras, A. N.: Transform domain watermarking: adaptive selection of the watermark's position and length. In: Proceedings of Visual Communications and Image Processing, VCIP 2003, July 2003
3. Sarkar, A., Madhow, U., Chandrasekaran, S., Manjunath, B·S.: Adaptive MPEG-2 video data hiding scheme. In: Proceedings of SPIE Security, Steganography, and Watermarking of Multimedia Contents IX, January 2007
4. Liu, H., Huang, J., Shi, Y.Q.: DWT-based video data hiding robust to MPEG compression and frame loss. Int. J. Image Graph. **5**(1), 111–134 (2005). doi:10.1142/S0219467805001689
5. Zhang, J., Li, J., Zhang, L.: Video watermark technique in motion vector. In: Proceedings of XIV Symposium on Computer Graphics and Image Processing, pp. 179–182, October 2001
6. Bodo, Y., Laurent, N., Dugelay, J.-L.: Watermarking video; hierarchical embedding in motion vectors. In: IEEE Proceedings of International Conference on Image Processing, September 2003
7. Fang, D.-Y., Chang, L.-W.: Data hiding for digital video with phase of motion vector. In: IEEE Proceedings of International Symposium on Circuits and Systems (ISCAS), May 2006
8. Noorkami, M., Mersereau, R.M.: Towards robust compressed-domain video watermarking for H.264. Proc. SPIE **6072**, 489–497 (2006)
9. Cao, H., Zhou, J., Yu, S.: An implement of fast hiding data into H.264 bitstream based on intra-prediction coding. Proc. SPIE **6043**, 123–130 (2005)
10. Proefrock, D., Richter, H., Schlauweg, M., Mueller, E.: H.264/AVC video authentication using skipped macroblocks for an erasable watermark. Proc SPIE **5960**, 1480–1489 (2005)
11. Hu, Y., Zhang, C., Su, Y.: Information hiding based on intra prediction modes for H.264/AVC. In: IEEE International Conference on Multimedia and Expo (ICME), Beijing, China, July 2–5, 2007
12. Kapotas, S.K., Varsaki, E.E., Skodras, A.N.: Data hiding in H.264 encoded video sequences. In: IEEE International Workshop on Multimedia Signal Processing (MMSP), Chania, Greece, October 1–3, 2007
13. Kim, S.M., Kim, S.B., Hong, Y., Won, C.S.: Data hiding on H.264/AVC compressed video. In: Proceedings of ICIAR 2007. LNCS, vol. 4633, pp. 698–707 (2007)
14. Wiegand, T., Sullivan, G.J., Luthra, A.: Draft ITU-T Recommendation H.264 and Final Draft International Standard 14496-10 AVC, JVT of ISO/IEC JTC1/SC29/WG11 and ITU-T SG16/Q.6, Doc. JVT-G050r1, Geneva, Switzerland, May 2003
15. Reference, J.V.T.: Software version JM 14.0 http://iphome.hhi.de/suehring/tml/download/
16. Gonzalez, R.C., Woods, R.E.: Digital Image Processing, 3rd Edn., Prentice-Hall, Englewood Cliffs (2008)
17. Bender, W., Gruhl, D., Morimoto, N.: Techniques for data hiding. Technical Report, Massachusetts Institute of Technology Media Lab (1994)

Author Biographies

Spyridon K. Kapotas received an honors degree in Electrical Engineering from the Electrical Engineering Department, University of Patras, Patras, Greece, in 1994 and an MSc degree in Sound and Vibration from the Institute of Sound and Vibration, University of Southampton, UK, in 1995. He is currently working towards the Ph.D. degree in the School of Science and Technology, Hellenic Open University, Greece. His current research interests include video coding, video transcoding and DSP algorithms for embedded systems. He has published four technical papers in international conferences. Mr. Kapotas worked for 9 years as a senior firmware engineer in Atmel Corp. He is currently working as a senior software engineer in Bytemobile, Inc.

Athanassios N. Skodras received the BSc degree in Physics from Aristotle University of Thessaloniki, Greece, in 1980, the MEng degree in Computer Engineering and Informatics and the Ph.D. degree in Electronics, in 1986, both from University of Patras, Greece. Since 1986 he has been holding teaching and research positions at the Departments of Physics and Computer Engineering and Informatics of the University of Patras, and the Research Academic Computer Technology Institute, Patras, Greece. As of October 2002 he is Professor of Digital Systems and Head of Computer Science, School of Science and Technology, Hellenic Open University, Patras, Greece. During the academic years 1988–1989 and 1996–1997 he was a Visiting Research Scientist at the Department of Electrical and Electronic Engineering, Imperial College, London, UK. His current research interests include image and video coding, digital watermarking for IPR protection, fast transform algorithms, real-time digital signal processing and multimedia applications. He has published over 100 technical papers in journals and conference proceedings, authored or co-authored 4 books, 1 book chapter, and holds 2 international patents on image down-sizing in the compressed domain. Dr. Skodras serves as an Associate Editor for *IEEE Signal Processing Letters*, the Springer *Journal of Real-Time Image Processing*, and Elsevier *Pattern Recognition*, and as a reviewer for numerous journals and conferences. He also serves as Vice Chair of the IEEE Greece Section, Chair of the IEEE Greece CAS & SSC Chapters, Chair of the Greek Association of Image Processing and Digital Media, and Technical Coordinator of the WG6 on image and video coding of the Greek Organization for Standardization. He is the General Chair of the 16th IEEE Int. Conference on Digital Signal Processing (DSP2009). He is the co-recipient of the first place Chester Sall Award for the best paper in the 2000 *IEEE Transactions on Consumer Electronics*, and the recipient of the 2005 IEEE Circuits and Systems Chapter-of-the-Year Award. Dr. Skodras is a Chartered Engineer, Senior Member of the IEEE, and member of the IET, EURASIP and the Technical Chamber of Greece.