Data Hiding in H.264 Encoded Video Sequences

Spyridon K. Kapotas, Eleni E. Varsaki and Athanassios N. Skodras
Digital Systems and Media Computing Laboratory, School of Science and Technology
Hellenic Open University
Patras, Greece
{s.kapotas, e.varsaki, skodras}@eap.gr

***Abstract*— A new method for high capacity data hiding in H.264 streams is presented. The proposed method 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. 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. This fragile data hiding approach can be mainly used for content-based authentication and covert communication.**

Keywords*— *H.264, data hiding, covert communication, authentication

I. Introduction

The widespread 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 digital medium or the message can be text, audio, picture or video depending on the size of the message and the capacity of the cover.

Early video data hiding approaches were proposing still image watermarking techniques extended to video by hiding the message in each frame independently [1]. Methods such as spread spectrum are 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 DFT (Discrete Fourier Transform), the DCT (Discrete Cosine Transform), and the DWT (Discrete Wavelet Transform) domains are 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 [8-10] 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 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 skipped macroblocks are used to embed data.

The well established H.264/AVC video coding standard has various motion-compensation units in sizes of 16×16, 16×8, 8×16, 8×8, and sub8×8 [11]. For sub8×8, there are further four sub-partitions of sub8×8, sub8×4, sub4×8, and sub4×4. In this paper we propose a new data hiding scheme, which takes advantage of the different block sizes used by the H.264 encoder during the inter prediction, in order to hide the desirable data. The message can be extracted directly from the encoded stream without knowing the original host video. This method is best suited for content-based authentication and covert communication applications.

The rest of the paper is organized as follows. In Section II we describe the new scheme. In Section III we present the simulation results. In Section IV the message extractor is briefly described. In Section V some future considerations are discussed. Finally, in Section VI conclusions are drawn.

II. Proposed Data Hiding Scheme

The main blocks of the H.264 video encoder are depicted in Fig. 1. The Temporal Prediction block is responsible for the *inter prediction* of each inter frame. Our scheme intervenes in the inter prediction process in order to hide the data.

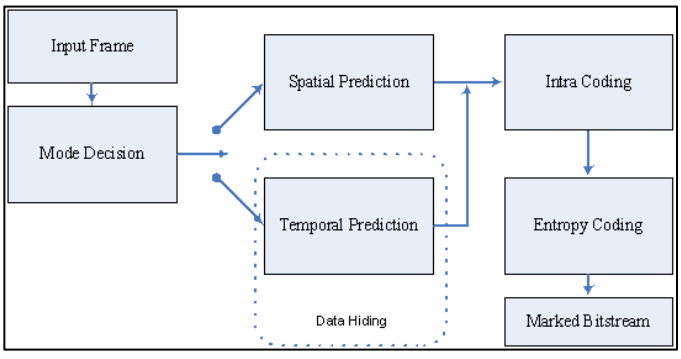

Figure 1. H.264 video encoder

The most important part of inter prediction is the motion estimation process, which aims at finding the "closest" macroblock (best match) in the previously coded frame for every macroblock of the current input frame. Then each macroblock, within the current frame, is motion compensated, i.e. its best match is subtracted from it, and the residual macroblock is coded. In order to increase the coding efficiency, the H.264 standard has adopted seven (7) different block types (16×16, 16×8, 8×16, 8×8, 8×4, 4×8, 4×4) and the motion estimation is applied on each of these types. The block type, which results in the best coding, is selected in the end. The basic idea of the proposed scheme is to force the encoder to choose a block type not in terms of coding efficiency, but according to our data hiding requirements.

First we assign a binary code to every block type according to Table I. For simplicity we use only 4 block types. That gives us 2 bits per block. Then we convert the embedding message into a binary number and we separate the bits in pairs. These pairs are mapped into macroblocks, which are going to be motion compensated, using the chosen block types as is shown in Fig. 2

TABLE I. BINARY CODES OF THE BLOCK TYPES

Block type	Binary code
16×16	00
16×8	01
8×16	10
8×8	11

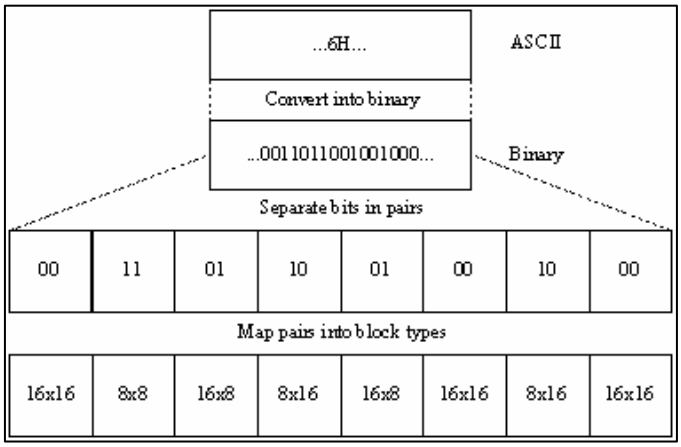

Figure 2. Message mapping into block types.

It is also important to define the data hiding parameters such as:

1. Starting frame: It indicates the frame from which the algorithm starts message embedding.
2. Starting macroblock: It indicates the macroblock within the chosen frame from which the algorithm starts message embedding.
3. Number of macroblocks: It indicates how many macroblocks within a frame are going to be used for data hiding. These macroblocks may be consecutive or even better, they may be spread within the frame according to a predefined pattern. Apparently, the more the macroblocks we use, the higher the embedding capacity we get. Moreover, if the size of the message is fixed, this number will be fixed, too. Otherwise it can be dynamically changed.
4. Frame period: It indicates the number of the inter frames, which must pass, before the algorithm repeats the embedding. This parameter is very important since it increases the possibilities of extracting the message even if some parts of the video sequence are missing. However, if the frame period is too small and the algorithm repeats the message very often, that might have an impact onto the coding efficiency of the encoder. Apparently, if the video sequence is large enough, the frame period can be accordingly large.

The encoder reads these parameters from a file. The same file is read by the software that extracts the message, so as both of the two codes to be synchronized.

Fig. 3 shows the block diagram of the proposed embedding algorithm. As an inter frame enters the Temporal Prediction module, the algorithm decides whether to use it for hiding a message or not, according to the hiding parameters. If the algorithm decides to use the frame, it chooses the macroblock candidates and performs the motion estimation on them, forcing the encoder to choose a specific block type according to the message mapping (Fig. 2). Then it lets the encoder to proceed with the encoding as in normal operation. In other words the algorithm fakes the motion estimation process, which the encoder would normally perform.

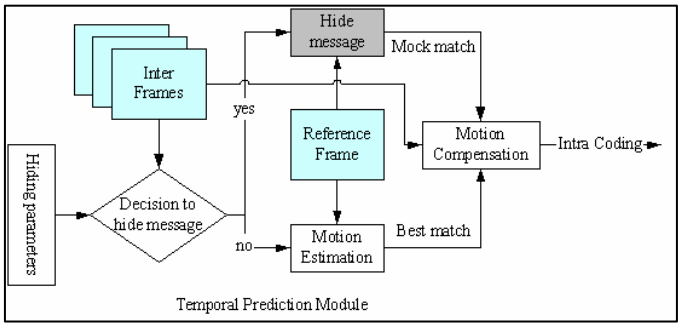

Figure 3. Block diagram of the proposed scheme

The proposed scheme may result in very high capacity proportional to the host video sequence size. Its major advantage is that it does not affect the visual quality of the video sequence and if the hiding parameters are properly controlled it does not affect the coding efficiency, either. In addition to that, it is extremely difficult for the decoder to detect the data hiding interference and this increases the invisibility of the hidden message. Finally, the message can be extracted directly from the encoded video stream without the need of the original host video sequence. Extended tests are conducted and useful conclusions are drawn in the following sections.

III. Simulation Results

The proposed scheme was integrated within version 11.0 of the reference JVT software [12]. The most important configuration parameters of the reference software are shown in Table II. The rest of the parameters have retained their default values. Note that the inter prediction optimizing parameters are disabled for simplifying the algorithm implementation.

TABLE II. CONFIGURATION PARAMETERS OF THE ENCODER

Profile	Baseline
Frames	100
Frame Rate	30 fps
Number of reference frames	10
Motion Estimation Algorithm	Full Search
RD Optimization	Disabled
8x8 Sub-blocks	Disabled
Rate Control	Disabled

Several video sequences in QCIF format were tested. Fig. 4, shows the PSNR results of each luma inter frame for the *Foreman* sequence. We refer to the inter frames since the message is inserted into these frames only. By default the H.264 encoder regards only the first frame as an intra and the rest as inter frames. The first intra frame has been excluded from Fig. 4.

From the results we observe that the proposed scheme does not actually affect the PSNR of the inter frames. This was expected since there is no bit rate constraint and thus our scheme does not provoke any loss of information. We would rather expect to see differences in the total bit-rate of the inter frames, due to the fact that the scheme interferes with the optimizing part of the inter prediction. Fig. 5 shows the bit rate differences of the inter frames between the original sequences and the marked ones. The bit-rate is generally increased proportionally to the capacity size.

Based on Fig. 4 and 5 we can assume that if we put a bit rate constraint on the encoder we should expect a PSNR decrease. Fig. 6 shows the PSNR differences of the inter frames between the original sequences and the marked ones when we enforce a 40 kbps bit rate constraint on the encoder. A maximum of 1.4 dB difference is experienced.

The small bit-rate reduction and the PSNR increase that we see in some cases in Fig. 5 and Fig. 6 respectively, are partly due to the stochastic choice of the message and mainly to the fact that the optimizing parameters of the encoder were disabled, in the sense that the encoder was not able to perform the best possible inter prediction during its normal operation.

The proposed scheme should ideally affect both the PSNR and the bit rate as less as possible. A few approaches, which may result in a great improvement of the proposed scheme, are discussed in Section V.

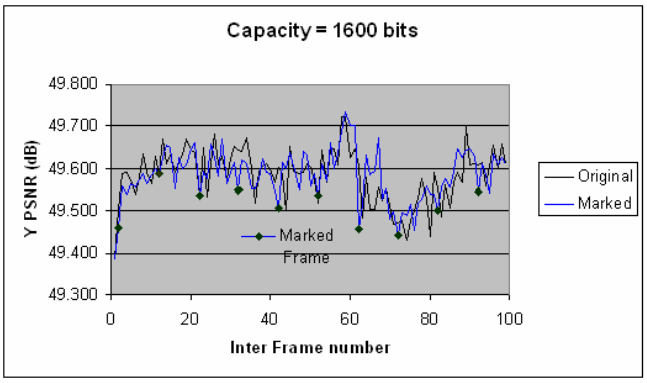

Figure 4. Foreman–PSNR of the luma inter frames

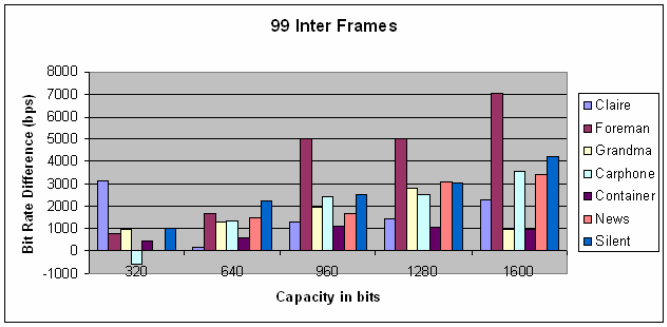

Figure 5. Bit rate differences of the luma inter frames

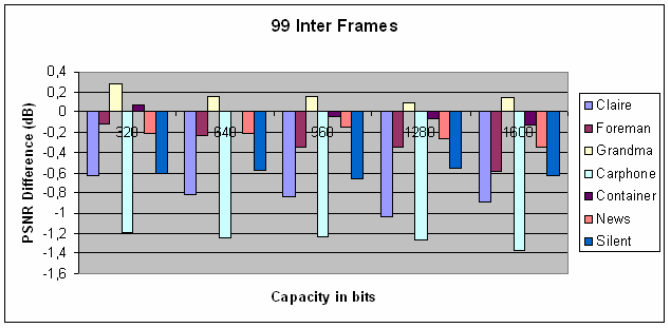

Figure 6. PSNR differences of the luma inter frames

IV. Message Extractor

The message extractor is software, not necessarily an H.264 decoder, which extracts the hidden message from the marked H.264 bitstream. The message extractor needs to partially decode the bitstream in order to discover the chosen block type of each macroblock of each inter frame. Then, it can form the hidden message according to Table I. Apparently, the message extractor must be aware of the hiding parameters, which were used by the encoder.

V. Future Work

In our current scheme we used only 4 different block types, namely 16×16, 16×8, 8×16, 8×8. However, the scheme can also use the sub partitions of the 8×8 type (8×4, 4×8, 4×4), thus increasing the available bits for coding to 8. Apparently,

the additional bits will increase the data capacity decreasing the number of the "twisted" macroblocks at the same time. Moreover, the scheme used consecutive macroblocks within a single frame in order to hide the data. Another improvement would have been if the macroblocks were spread within the frame or even better if the macroblocks were spread within multiple frames. This approach would improve the coding efficiency, since the "motion error", which is produced by the scheme, will not be accumulated in one place. In addition to that, the assignment of the binary codes in Table I could be modified so as to take into account some video statistics. For example the 16x16 block type appears more often than the other types. The message can therefore be coded using a Huffman coding and the Huffman code with the highest probability could be assigned to the 16x16 block type. The gain of this approach will be that our scheme will most likely choose the block type, which would have been chosen by the encoder.

VI. CONCLUSIONS

In this paper we present a new data hiding scheme for H.264 encoded video sequences. Embedding takes place during the encoding process and utilizes the advanced inter prediction features of the H.264 encoder. Its main advantage is that it is a blind scheme and its affect on video quality or coding efficiency is almost negligible. It is highly configurable, thus it may result in high data capacities. Finally, it can be easily extended, resulting in better robustness, better data security and higher embedding capacity.

ACKNOWLEDGEMENTS

This work was funded by the European Union - European Social Fund (75%), the Greek Government - Ministry of Development - General Secretariat of Research and Technology (25%) and the Private Sector in the frames of the European Competitiveness Programme (Third Community Support Framework - Measure 8.3 programme PENED-contract no.03ED832).

REFERENCES

[1] J. J. Chae, B. S. Manjunath, "Data Hiding in Video", IEEE Proc. Int. Conf. on Image Precessing, pp.243-246, 1999.

[2] V. Fotopoulos, A. N. Skodras, "Transform Domain Water-marking: Adaptive Selection of the Watermark's Position and Length", Proc. Visual Communications and Image Processing, VCIP2003,,July 2003.

[3] A. Sarkar, U. Madhow, S. Chandrasekaran, B. S. Manjunath, "Adaptive MPEG-2 Video Data Hiding Scheme", Proc. SPIE Security, Steganography, and Watermarking of Multimedia Contents IX, Jan. 2007.

[4] H. Liu, J. Huang, Y. Q. Shi, "DWT-Based Video Data Hiding Robust to MPEG Compression and Frame Loss", Int. Journal of Image and Graphics Vol.5 No.1, pp. 111-134, Jan. 2005.

[5] J. Zhang, J. Li, L. Zhang, "Video Watermark Technique in Motion Vector", Proc. of XIV Symposium on Computer Graphics and Image Processing, pp.179-182, Oct.2001.

[6] Y. Bodo, N. Laurent, J.-L. Dugelay, "Watermarking Video, Hierarchical Embedding in Motion Vectors", Proc. Int. Conference on Image Processing, Sept. 2003.

[7] D.-Y. Fang, L.-W. Chang, "Data Hiding for Digital Video with Phase of Motion Vector", IEEE Proc. Int. Symposium on Circuits and Systems, ISCAS 2006, May 2006.

[8] M. Noorkami, R. M. Mersereau, "Towards Robust Compressed-Domain Video Watermarking for H.264", Proc. SPIE, Vol. 6072, pp. 489-497, 2006.

[9] H. Cao, J. Zhou, S. Yu, "An Implement of Fast Hiding Data into H.264 Bitstream based on Inter-Prediction Coding", Proc. SPIE, Vol. 6043, pp. 123-130, 2005.

[10] D. Proefrock, H. Richter, M. Schlauweg, E. Mueller, "H.264/AVC Video Authentication Using Skipped Macroblocks for an Erasable Watermark", Proc. SPIE Vol. 5960, pp. 1480-1489, 2005.

[11] T. Wiegand, G. J. Sullivan, A. Luthra, "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.

[12] JVT Reference Software version JM 11.0 http://iphome.hhi.de/suehring/tml/download/old_jm/