Data Hiding in Compressed Video using LSB Algorithm

Pradip N. Ghorpade
Dept. of Electronics & Telecommunication
Bhivarabai Sawant College of Engg & Research, Narhe
Pune, India

D. V. Jadhav
Dept. of Electronics & Telecommunication
Bhivarabai Sawant College of Engg & Research, Narhe
Pune, India.

***Abstract*—This paper deals with data hiding in motion vectors of compressed video using Least Significant Bit algorithm. Data hiding is different technique than cryptography. Data hiding embed the data in to digital media for the identification, annotation, and copyright. Data hiding is secure adding of information content to a data product, using mathematical techniques. These mathematical techniques work at the primitive level of digital data products, with no perceptual degradation to data product integrity. We target the motion vectors to encode and reconstruct the forward predictive (P)-frame. The choice of these motion vectors are based on their associated macroblock prediction error, which is different from the approaches based on the motion vector attributes. The secret message is embedded in the least significant bit of the candidate motion vectors. Data hiding in motion vectors relies on changing the motion vectors based on their attributes such as their magnitude, phase angle, etc. The method is implemented & tested for hiding data in sequences of groups of pictures & the results are evaluated. Experimental result shows that the proposed method is efficient for data hiding in compressed video by using LSB algorithm.**

***Index Terms*—Data hiding, motion vectors, New Three Step Search (NTSS), LSB Algorithm, motion estimation.**

I. Introduction

Data hiding is process of secretly embedding information inside a data source without changing its perceptual quality. Steganography is the art of concealing a message, image or file within another message, image or file. The main goal of steganography is to hide a message m in some audio or video (cover) data *d*, to obtain new data d^h, practically indistinguishable from *d*, by people, in such a way that an eavesdropper cannot detect the presence of m in d^h. Steganography methods usually do not need to provide strong security against removing or modification of the hidden message. The block diagram of the data hiding technique is shown in Fig. 1.

We target the internal dynamics of video compression, specifically the motion estimation stage. We have chosen this stage because its contents are processed internally during the video encoding/decoding which makes it hard to be detected by image steganalysis methods and is lossless coded. When the frame rate is sufficiently high, there is a great amount of similarity between successive frames. It is more efficient to code the difference between frames than the frames themselves. Motion estimation is aimed to reduce such strong temporal redundancy present in video sequence. So far, the most successful technique for motion estimation is perhaps the block matching algorithm (BMA) which has been adopted by various video coding standards, such as H.261, H.263, MPEG-1, MPEG-2 and MPEG-4. The block matching is constrained to search within the selected sector for a magnitude to be larger than the predefined threshold.

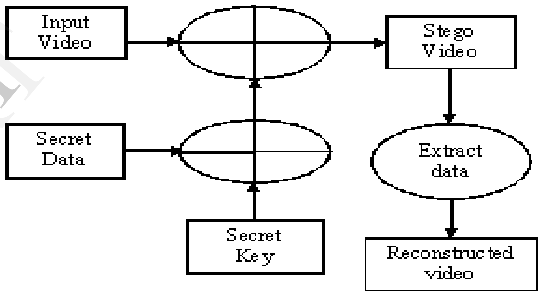


Fig. 1. Block diagram of data hiding technique

In the literature, most work applied on data hiding in motion vectors relies on changing the motion vectors based on their magnitude, phase angle, etc. In [3] and [4], the data bits of the message are hidden in some of the motion vectors whose magnitude is above a predefined threshold, and are called candidate motion vectors (CMVs). A single bit is hidden in the least significant bit of the larger component of each CMV. The authors in [5] and [6] embed the data in video using the phase angle between two consecutive CMV. These CMV are selected based on the magnitude of the motion vectors as in [3].

The rest of the paper is organized as follows: in section II we overview the terms of video compression and decompression. In section III we have given problem definition. New three step-search method is given in section IV. The proposed method used here is given in section V followed by the results and analysis in section VI. Finally the conclusions are given in section VII.

II. Background and Notations

In this section, we overview lossless video compression to define our notation and evaluation metrics. The I-frame is used as a reference frame for encoding a group of forward predicted (P)-or bidirectionally predicted (B)-frames. In the commonly used Motion Picture Export Group (MPEG-2) standard [7], the video is group of pictures (GOPs) whose frames can be encoded in the sequence: [I,B,B,P,B,B,P,B,B]. The temporal redundancy between frames is exploited using block based motion estimation that is applied on macroblocks $\mathcal{B}$ in P or B and searched in target frames.

Generally the motion field in video compression is assumed to be translational with horizontal component d^x and vertical component d^y and denoted in vector form by d(x) for the spatial variables x=(x, y). Motion vectors can be obtained using methods such as new three step search, three step search etc.; this is based on the required compression ratio & the reconstruction quality. Since d does not represent the true motion in the video then the compensated frame $\dot{P}$ using (x + d(x)) must be associated with a prediction error $E(x) = (P - \dot{P})(x)$ in order to be able to reconstruct $P = \dot{P} + E$ with minimum distortion at the decoder in case of a P-frame. The motion vectors d are lossless coded and thus become an attractive place to hide a message that can be blindly extracted by a special decoder.

The decoder receives the encoded pair (d, $\tilde{E}$), applies motion compensation to form $\dot{P}$ or $\dot{B}$ and decompresses $\tilde{E}$ to obtain a reconstructed E_r. Since E and E_r are different due to quantization effect, then decoder unable to reconstruct P identically but it alternatively reconstructs $P_r = \dot{P} + E_r$. The reconstruction quality is measured by the mean squared error (MSE) $P - P_r$, represented as peak signal-to-noise ratio (PSNR) denoted by $\mathcal{R}$.

III. Problem Definition

Data hiding in motion vectors at the encoder replaces the regular pair (d, $\tilde{E}$), to become (d^h, $\tilde{E}^h$), where h denotes hiding. We define data hiding in motion vectors of compressed video in the context of super-channel [8]; the secret message m is hidden in the host video signal x = (d, E) to produce the composite signal s = (d^h, E^h). The composite signal is subject to video lossless compression to become y = (d^h, $\tilde{E}^h$). The message m can be identically extracted from y. This robustness constrain should have low distortion effect on the reconstructed video as well as low effect on the data size (bit rate). The cost function mean squared error is given by equation (i)

$$MSE = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(C_{ij} - R_{ij}\right)^2 \qquad \text{(i)}$$

Where N is the size of the micro block, C_{ij} and R_{ij} are the pixels being compared in current macro block and reference macro block, respectively. Peak-Signal-to-Noise-Ratio (PSNR) given by equation (ii) characterizes the motion compensated image that is created by using motion vectors and macro blocks from the reference frame.

$$\text{PSNR} = 10Log_{10}\left[\frac{(\text{Peak to peak value of the original data})^2}{MSE}\right] \qquad \text{(ii)}$$

The selection of the CMV is the key difference between different methods. For instance [3] and [4] choose the CMV based on their magnitude. On the other hand [5] and [6] choose CMV based on their magnitude and the phase between consecutive motion vectors. In our proposed method we rely directly on the associated macroblock prediction error, such that we choose our CMV associated with macroblocks of high prediction error. If we temper with these CMVs, then we will not have poor effect on the video reconstruction quality.

IV. New Three Step Search

New three step search used a halfway-stop technique to speed up the (quasi-) stationary blocks matching. The NTSS performs better than TSS in terms of mean square error and the number of search points. For the search window w = 7, in the first step the NTSS algorithm utilizes a 3 × 3 grid search pattern at the center in addition to the larger 9 × 9 grid in the TSS. If the minimum block distortion measure (BDM) point is one of the eight points on the 3 × 3 grid, only additional five or three points will be checked, otherwise the search window center will be moved to the winning point and the following procedure is the same as in TSS. The details are as follows:

Step 1) Totally 8 + 9 = 17 points are checked including the central nine points on the 3 × 3 grid and the eight neighboring points on the 9 × 9 grid. If the minimum BDM point is the search window center, the search will be terminated; otherwise go to step 2.

Step 2) If one of the central eight neighboring points on the 3 × 3 grid is found to be the minimum in the first step, go to step 3; otherwise go to step 4.

Step 3) Move the small 3 × 3 search window so that the window center is the winning point found in step 1. Search additional five or three points according to the location of the previous winning point, then the search will stop.

Step 4) Reduce the large 9 × 9 search window size by half and move the center to the minimum BDM point in step 1, follow the searching process of step 2 and step 3 in TSS.

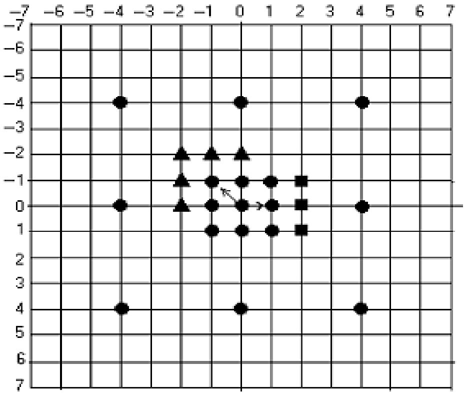


Fig. 2. Two different search paths for finding MV within 5 × 5 area in NTSS

Fig. 2 shows two different search paths for finding motion vector within 5 × 5 area. According to the halfway-stop technique, the NTSS needs 17 search points for stationary blocks and 20 or 22 points for small motion vectors within central 5 × 5 area. For the worst case, 25 + 8 = 33 search points will be required compared to 25 points needed in TSS.

V. Proposed Method

Our data-hiding algorithm is applied at the encoder side, uses the regular pair (d, $\tilde{E}$) produced, tampers d to become d^h, and thus replaces them by the pair (d^h, $\tilde{E}^h$) for each P-frame in the group of picture (GOP) as shown in Procedure 1. The secret message is organized as a bitstream m (k), 0 < k ≤ K: message length. A subset of d is selected to be the CMV $\dot{d}$. The least significant bit (LSB) of both components $\dot{d}^x$, $\dot{d}^y$ are replaced by bits of the message.

***Procedure 1** Data hiding in GOP*

Input: Secret data m, GOP (d, $\tilde{E}$), k
Output: Stego video
Step 1: Read the input Video
Step 2: Perform frame separation
Step 3: Apply Integer DCT on each 8×8 block
Step 4: Perform Zigzag Scanning on each 8×8 block
Step 5: Apply Huffman coding to compress the frame
Step 6: Apply secret key to hide the data
Step 7: Apply LSB Algorithm to embed data
Step 8: Generate Stego video

The decoder receives the pair (d^h, $\tilde{E}^h$) and it can decode d^h without loss and decompresses $\tilde{E}^h$ to obtain a lossy reconstructed version E_r^h. During normal operation for viewing the video, the decoder is able to reconstruct P_r^h by suitable compensation from reference frames using (x = d^h(x)) and adding E_r^h to it. Acting as a new kind of motion estimation, procedure 1 will have two effects on the new compressed video: change in data size and reconstruction quality. The data extractor operates to extract the hidden message as a special decoder and our method is straightforward as shown in Procedure 2. After data extraction from the consecutive GOPs the hidden data is reconstructed back by concatenation of the extracted bitstream.

***Procedure 2** Data extraction from GOP*

Input: GOP (d^h, $\tilde{E}^h$), k
Output: Secret data m
Step 1: Read Stego video
Step 2: Perform decoding using IDCT and Inverse Huffman Coding
Step 3: Extract hidden data using ILSB and Secret Key.

LSB Algorithm

Embedding message is performed for two pixels *X* and *Y* of a cover image at a time and then adjusting one pixel of the (*X*, *Y*) to embed two secret bits message *s1 s2*. The embedding flowchart is shown in Fig. 3 and the embedding procedure is described as following:

Step 1: If the LSB of *X* is the same as *s1*, go to step 2. Otherwise, go to step 3.
Step 2: If the value of *f* (*X, Y)* is the same as *s2*, do not change any pixel. Otherwise, the value of pixel *Y* is increased or decreased by 1.
Step 3: If the value of *f* (*X* −1, *Y) is* the same as *s2*, the value of pixel *X* is decreased by 1.

Otherwise, the value of pixel *X* is increased by 1. Where the function *f*(*X*, *Y*) is defined as,

$$F\left(X',Y'\right) = LSB\left[\frac{x'}{2}\right] + Y' \qquad \text{(iii)}$$

Since this new LSB matching method just only increase or decrease 1 in two adjacent pixels, the difference of the two neighborhood pixel between cover image and Stego-image is very small. Hence, it can keep high quality while hiding data.

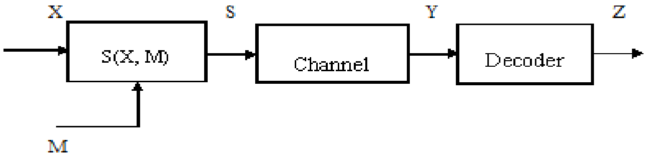


Fig. 3. General information embedding problem model

The simple problem model of Fig. 3 captures most of their fundamental features [8]. We wish to embed some digital information or watermark m in some host signal vector x. This host signal could be a vector of pixel values or Discrete Cosine Transform (DCT) coefficients from an image, for example. Alternatively, the host signal could be a vector of samples or transform coefficients, such as Discrete Fourier Transform (DFT) or linear prediction coding coefficients, from an audio or speech signal. An embedding function maps the host signal x and embedded information m to a composite signal. The embedding should not unacceptably degrade the host signal, so we have some distortion measure *D* between the composite and host signals. This model [8] is sufficiently general to include both random and deterministic, and both signal-independent and signal dependent, perturbation vectors. The decoder forms an estimate Om of the embedded information m based on the channel output y. The robustness of the overall embedding-decoding method is characterized by the class of perturbation vectors over which the estimate is reliable.

VI. Experimental Result

We implemented the data hiding and extraction Algorithms for video. The parameters of our experiments are macro block of size b = 16, motion vector representation bits n = 8. We used new three-step search motion estimation algorithm and LSB algorithm for embedding secret data into input video. The video sequence is organized into consecutive GOP as [I,B,B,P,B,B,P,B,B]. The compression to the I-frame and the prediction error of the P-frames are implemented using JPEG compression. We tested our algorithms on three test sequences: outdoor1, outdoor2 and outdoor3 which are all shown in Fig. 4. All the outdoor1, outdoor2 and outdoor3 sequence have frame size of 352 × 400 which corresponds to 550 macroblocks $\mathcal{B}$ per frame. The number of macroblocks per frame and the total number of frames for each sequence are given in the first column of Table I. The motion estimation method which is used here is given in the second column of Table I. Analysis of PSNR values of original information (secret) image versus extracted image.

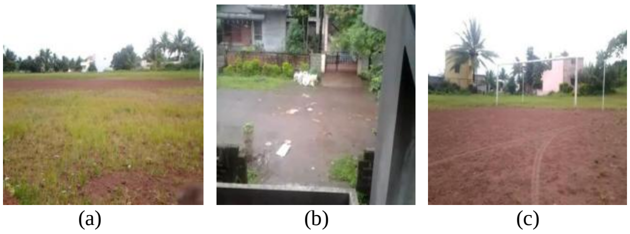


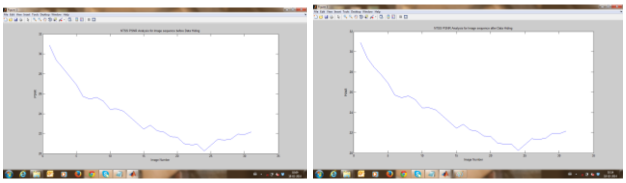
Fig. 4. First image of the test sequences. (a) Outdoor1; (b) Outdoor2; (c) Outdoor3.

The sequences outdoor1 and outdoor3 have high motion dynamics while outdoor2 have moderate finger motion at the foreground. We evaluated this algorithm and compared PSNR value of original information image versus extracted image for different input videos.

TABLE I
Analysis of PSNR values for different test sequences

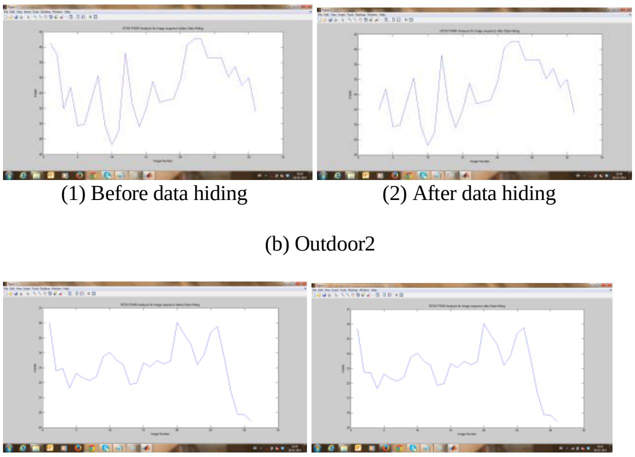
Experiment Setup		**PSNR values of original information image Vs extracted image**
Test sequence	Motion estimation search method	
Outdoor1 (550B /frame) (33 frames)	NTSS	7.1472
Outdoor2 (550 B /frame) (33 frames)	NTSS	13.9482
Outdoor3 (550 B /frame) (33 frames)	NTSS	5.8936

We kept compression ratio constant for all three test sequences. Also from Table I it is observed that PSNR value obtained from test sequence outdoor2 is better than outdoor1 and outdoor3. Fig. 5 shows NTSS PSNR analysis for three test sequence before data hiding and after data hiding.



(1) Before data hiding (2) After data hiding

(a) Outdoor1



(1) Before data hiding (2) After data hiding

(b) Outdoor2

(1) Before data hiding (2) After data hiding

(c) Outdoor3

Fig. 5. NTSS PSNR analysis for different test sequences

VII. Conclusion

We proposed a method for data-hiding in motion vectors of compressed video. We selected those motion vectors whose associated macroblocks prediction error is high (low PSNR) for hiding a bit in each of their horizontal and vertical components. The hiding and extraction algorithms are implemented and the results are evaluated. The proposed method is found to have lower distortion to the quality of the video.

References

[1] Hussein A. Aly, “Data Hiding in Motion Vectors of Compressed Video Based on Their Associated Prediction Error”, IEEE Trans. on Information Forensics and Security, vol. 6, no. 1, March 2011.

[2] Xuan Jing and Lap-Pui Chau, “An efficient three-step search algorithm for block motion estimation”, IEEE Trasn. on Multimedia, vol. 6. No. 3, June 2004.

[3] C. Xu, X. Ping, and T. Zhang, “Steganography in compressed video stream,” in Proc. Int. Conf. Innovative Computing, Information and Control (ICICIC’06), 2006, vol. II, pp. 803-806.

[4] F. A. P. Petitcolas, R. J. Anderson, and M. G. Kuhn, “Information hiding — A survey,” Proc. IEEE, vol. 87, no. 7, pp. 1062–1078, Jul. 1999.

[5] D.-Y. Fang and L.-W. Chang, “Data hiding for digital video with phase of motion vector ”in Proc. Int. Symp. Circuits and Systems (ISCAS), 2006, pp. 1422–1425.

[6] X. He and Z. Luo, “A novel steganographic algorithm based on the motion vector phase “in Proc. Int. Conf. Comp. Sc. and Software Eng., 2008, pp. 822–825.

[7] Generic Coding of Moving Pictures and Associated Audio Information: Video, 2 Edition, SO/IEC13818-2, 2000.

[8] B. Chen and G. W. Womell, “Quantization index modulation for digital watermarking and Information embedding of multimedia,” J. VLSI Signal Process., vol. 27, pp. 7-33, 2001.