Shanableh, T. (2012). Data hiding in MPEG video files using multivariate regression and flexible macroblock ordering. IEEE transactions on information forensics and security, 7(2), 455-464.
DOI: 10.1109/TIFS.2011.2177087

Data hiding in MPEG video files using Multivariate Regression and Flexible Macroblock Ordering

Tamer Shanableh

Department of Computer Science and Engineering
American University of Sharjah, UAE
Fax: +971 6 515-2979
tshanableh@aus.edu

Abstract

This paper proposes two data hiding approaches using compressed MPEG video. The first approach hides message bits by modulating the quantization scale of a constant bitrate video. A payload of one message bit per macroblock is achieved. A second order multivariate regression is used to find an association between macroblock level feature variables and the values of a hidden message bit. The regression model is then used by the decoder to predict the values of the hidden message bits with very high prediction accuracy. The second approach uses the flexible macroblock ordering feature of H.264/AVC to hide message bits. Macroblocks are assigned to arbitrary slice groups according to the content of the message bits to be hidden. A maximum payload of three message bits per macroblock is achieved. The proposed solutions are analyzed in terms of message extraction accuracy, message payload, excessive bitrate and quality distortion. Comparisons with previous work reveal that the proposed solutions are superior in terms of message payload whilst causing less distortion and compression overhead.

Keywords: Data hiding; Steganography; MPEG coding; multivariate regression; flexible macroblock ordering

EDICS: WAT-STEG.

1. Introduction:

Data hiding techniques can be used to embed a secrete message into a compressed video bit stream for copyright protection, access control, content annotation and transaction tracking. Such data hiding techniques can also be used for other purposes. For instance, [1] used data hiding techniques to assess the quality of compressed video in the absence of the original reference. The quality is estimated based on computing the degradations of the extracted hidden message. The authors of [2] used data hiding to enable real time scene change detection in compressed video. The information is hid using the motion compensation block sizes of an H.264/AVC video. Data hiding is also used for error detection and concealment in applications of video transmission. Edge orientation information and number of bits of a block are hidden in the bit stream for that purpose [3].

In general, the existing solutions rely on hiding message bits in DCT coefficients, Motion Vectors (MVs), quantization scale or prediction modes.

Examples of data hiding using DCT coefficients include the use the parity of the quantized coefficients to hide a message [4]. Additionally, [5] utilized zero-length codes to insert a dummy value at certain locations to indicate message bits.

Examples of using MVs for data hiding include [6], where phase angles of MVs are used to hide messages. The work in [7] and [8] on the other hand, proposed solutions for using the magnitude of MVs for data hiding. More specifically, [8] uses the least significant bit of both components of candidate motion vectors to embed a secret message. The candidate motion vectors are selected based on the prediction error of the underlying macroblock. MVs associated with high prediction errors are chosen. A prediction error threshold is computed per frame and transmitted in the video bit stream to guide the decoder in recognizing the MVs that carry bits of the secret message.

The quantization scale is also used for data hiding, a recent publication in [9], proposed to divide the quantization scale of a macroblock by a certain factor. The factor is multiplied by all AC coefficients

in the corresponding macroblock. The procedure is referred to as promoting and exiting a macroblock. If a message bit to hide is equal to zero then such a procedure is followed otherwise no action is taken.

From a syntax viewpoint, since a relatively large number of prediction modes and block sizes are available in H.264/AVC, it has been proposed to use these variants to hide message bits.

Likewise the work in [10] proposed the use of intra prediction modes to hide message bits. It was shown that 1 bit can be hidden in each candidate 4x4 intra block. Additionally, the work in [11] utilized the block types and modes of intra-coded blocks of H.264/AVC to hide message bits.

Data hiding can also be applied prior to compression. For example, [12] introduced a method that is robust to heavy JPEG compression. It is also possible to hide data in the wavelet domain as reported in [13]. In such an approach, significant wavelet coefficients are identified and used for embedding a message payload. Lastly, hiding of data can also be applied in the compressed domain. For example, the work in [14] proposed hiding messages in the compressed H.264/AVC I-frames without the introduction of drift distortion.

Steganalysis on the other hand, is the process of detecting the presence of hidden messages in multimedia. Steganalysis can be applied to digital images and to digital video as reported in [15] and [16] respectively. Existing work on video-based steganography takes such analysis into account and tries to maintain the statistics of carrier before and after message hiding. For example the work in [17] proposed a sub-histogram preserving approach for quantization modulation using matrix encoding.

In this paper we propose two novel solutions for data hiding. In first solution, the message bits are hidden by modifying the quantization scale of MPEG video coded with constant bit rates. Features are extracted from individual macroblocks and a second order regression model is computed. The decoder uses the regression model to predict the content of the hidden message based on macroblock-level feature variables. In the second solution, both constant and variable bit rate coding

are supported. The solution utilizes the Flexible Macroblock Ordering (FMO) feature of H.264/AVC video for message hiding and extraction. It is shown that both solutions can hide messages at an average payload of around 10kbit/s and 30kbit/s respectively. Therefore, the applications of such solutions are not restricted to copyright protection where few bits are hidden per frame. Rather, the proposed solutions can be used for other applications such as content annotation, transaction tracking, error detection and error concealment.

This paper is organized as follows. Section 2 introduces message hiding using quantization scale modulation and multivariate regression. Section 3 introduces message hiding using FMO. Experimental results and comparisons with existing work are reported in Section 4. Lastly, Section 5 concludes the paper.

2. Message hiding using quantization scale modulation:

To hide a message using quantization scale modulation, the message is first converted into a binary stream of bits. During the MPEG encoding of individual macroblocks, the message bits are read one at a time. For each coded macroblock, the quantization scale is either incremented or decremented based on the corresponding message bit. Clearly, if the original quantization scale was either the lowest or largest allowable values then no modification is applied. This simple process of hiding a message bit in a macroblock is illustrated in Figure 1.

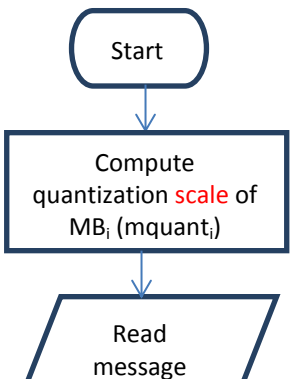


Figure 1. Message insertion flowchart for one macroblock

Although the message hiding procedure is straightforward, nonetheless, the question that remains is how to extract the message from the bistream. This problem can be solved by extracting macroblock-level feature variables during the encoding process. Once the whole message is hidden we end up with a feature matrix and a message vector. We will then treat the feature matrix as predictors and the message bits as a response variable and use multivariate regression to compute a prediction model. Once computed, the prediction model can be used to predict the message bit hidden in a given macroblock based on its feature variables. In the following sections we elaborate on the extraction of macroblock features from an MPEG-2 video, consequently, we formulate the message extraction as a regression problem.

2.1 Macroblock level features variables:

The following feature variables are extracted or computed from a MPEG-2 video stream for each coded macroblock:

1. The first feature is the virtual buffer discrepancy from uniform distribution model. This discrepancy is computed using Equation (1):

$$d_j^t = d_0^t + B_{j-1} - \left(\frac{T_t * (j-1)}{\#MBs}\right) \qquad (1)$$

Where the subscript *j* indicates a macroblock index, *#MBs* indicates the total number of macroblocks in a video frame and *t* indicates the frame type; I, P or B.

d_0^t is the initial buffer fullness at the beginning of coding a frame. It is calculated as the accumulated differences between the actual number of coded frame bits minus the target number of frame bits. d_0^t is updated after the encoding of each video frame. Additionally, B_{j-1}indicates the number of bits spent on coding the previous macroblocks in the current frame. Lastly, T_t indicates the target number of bits in the current Group of Pictures (GoP). The computation of which depends on the overall bitrate and frame rate, it also depends on number of bits used for coding the previous frames in the same GoP, the remaining number of P and B frames in the current GoP and the average quantization scale of the previous frames in the same GoP.

It can be concluded that the virtual buffer discrepancy from uniform distribution model can be recalculated at the decoder for each macroblock. Note that the video bitrate, the frame rate, horizontal and vertical image size are all part of the video sequence header. Hence, provided that the GoP structure is known, the decoder can use this information and keep track of the number of bits spent on previous frames and previous macroblocks to compute the value of the virtual buffer discrepancy. Assuming that the GoP structure is unknown, which is unlikely, the bit stream can be scanned ahead of computing the virtual buffer discrepancy to figure out the total number of P and B frame in a GoP.

2. The second feature is the spatial activity of the underlying macroblock. This activity is computed from the 4 original (i.e. non-coded) luminance blocks of the current macroblock. It is computed using Equation (2):

$$act_j = 1 + \min(v_{b1}, v_{b2}, v_{b3}, v_{b4}) \qquad (2)$$

Where the subscript *j* indicates a macroblock index. The variables $v_{b1}, v_{b2}, v_{b3}, v_{b4}$ indicate the spatial variance of each luminance block in a frame-based coding.

The encoder uses this spatial activity to adaptively modify the value of the quantization scale according to the spatial activity of the current macroblock. However since the variance is calculated using the pixel values of the original frame, as opposed to the reconstructed frame, this spatial activity measure is estimated at the decoder using calculation based on reconstructed frame instead. The consequences of which are elaborated upon in the experimental results section.

3. The third feature is the actual quantization scale of the current macroblock. This scale is available from the macroblock header in the video bit stream.

These feature variables are used in the system training and prediction of the hidden message as explained on the next section.

2.2 Message prediction

The message prediction problem is formulated using a second order multivariate regression. The response variable in this case is the message binary bits denoted by the vector **m.**

As mentioned previously, each macroblock has 3 feature variables, consequently, the predictors or the feature vectors of *n* macroblocks are arranged into one matrix which is referred to as the feature matrix. This matrix is denoted by **X** as shown in Equation (3)

$$\mathbf{X} = \begin{bmatrix} \boldsymbol{x_{1,1}} & \boldsymbol{x_{2,1}} & \boldsymbol{x_{3,1}} \\ \vdots & \vdots & \vdots \\ \boldsymbol{x_{1,n}} & \boldsymbol{x_{2,n}} & \boldsymbol{x_{3,n}} \end{bmatrix} \qquad (3)$$

The subscripts of the matrix elements $x_{j,i}(j = 1..3, i = 1..n)$ indicate the index of feature variables and the number of macroblocks, respectively.

To perform a nonlinear mapping between the predictors or the feature matrix **X** and the response variable **m,** the dimensionality of the rows or the feature vectors in matrix **X** is expanded into an r^{th} order. One approach to expanding the dimensionality is the reduced model polynomial expansion [18]. We refer to the expanded feature matrix as $\mathbf{P} \in \mathcal{R}^{nxk}$ where k is the dimensionality of the expanded feature vectors. According to [18], the dimensionality of the expanded feature vector is defined by $k = 1+r + l(2r-1)$. Where l denotes the number of features variables (l=3 in our case). In this work we use a second order expansion hence, k is equal to 12.

The second order expanded terms consist of the following values:

$$\begin{aligned} & p(x) = [1, \\ & x_j^k \mid 1 \le j \le l \ \wedge \ 1 \le k \le r, \\ & (x_1 + x_2 + \cdots + x_l)^k \mid 1 \le k \le r, \\ & x_j\left(x_1 + x_2 + \cdots + x_l\right)^{k-1} \middle| 1 \le j \le l \ \wedge \ 2 \le k \le r] \end{aligned} \tag{4}$$

For example, if a feature vector contains 2 variables only [x_1, x_2] then the expanded feature vector shall contain the following terms: $\left[1, x_{1,}, x_2, x_1x_2, x_1^2, x_2^2, (x_1 + x_2), (x_1 + x_2)^2, x_1(x_1x_2), x_2(x_1x_2)\right]$.

The mapping between **P** and **m** is achieved by using least-squared error objective criterion of Equation (5):

$$\boldsymbol{\alpha}^{opt} = \arg_{\boldsymbol{\alpha}} \min \|\mathbf{P}\boldsymbol{\alpha} - \mathbf{m}\|_2 \tag{5}$$

Where $\|.\|_2$ denotes the L_2 norm. Minimizing the objective function results in Equation (6):

$$\boldsymbol{\alpha}^{opt} = (\mathbf{P}^{\mathrm{T}}\mathbf{P})^{-1}\mathbf{P}^{\mathrm{T}}\,\mathbf{m} \tag{6}$$

Where $\boldsymbol{\alpha}^{opt} \in \mathcal{R}^{kx1}$. Therefore, using a second order expansion, the total number of regression weights needed is 12.

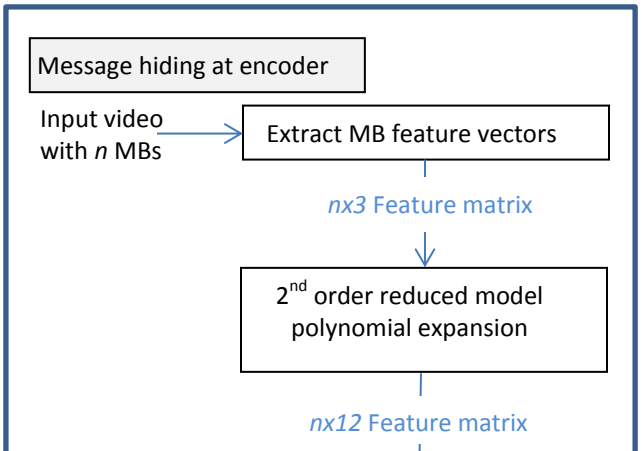
2.3 Message Extraction:

To extract the hidden message from a coded video, the feature variables of each macroblock are computed and/or extracted from the bitstream. The feature vectors are consequently arranged into a feature matrix and expanded to the second order, as illustrated in Equation (4) above, resulting in matrix **P**. The feature matrix is multiplied by the model weights $\boldsymbol{\alpha}^{opt}$ to generate the predicted hidden message $\hat{\mathbf{m}}$ as follows:

$$\hat{\mathbf{m}} = \mathbf{P} * \boldsymbol{\alpha}^{opt} \qquad (7)$$

The process of message hiding and prediction is summarized in Figure 2. Notice that the feature extraction and polynomial expansion steps are repeated at both stages of message hiding and prediction. As such, the feature vector need not be transmitted with the bitstream.

Following the assumption made in [8], one can assume that the model weights can be hidden in the video bitstream using any other existing message hiding techniques. The work in [8] computes a threshold for each and every predicted frame and stores them in the I frame of the underlying GoP. In our case however, we only need to transmit 12 weights for the whole video sequence. Although not implemented in this work, a potential hiding venue for the model weights can be the concealment motion vectors of the first I frame if MPEG-2 video is used.



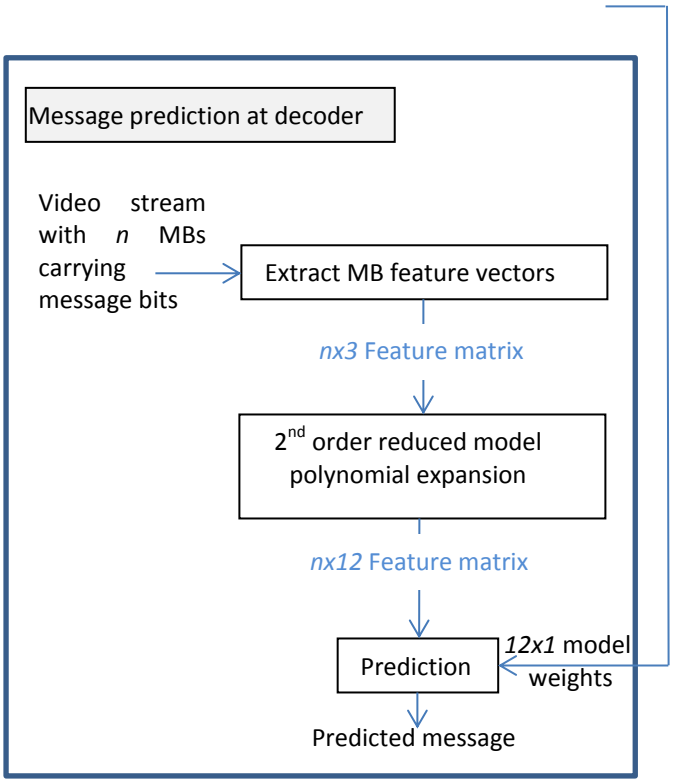


Figure 2. Block diagram of message hiding and prediction

For completeness, an example of message hiding using the proposed approach follows. A message is generated and a video sequence is encoded whilst hiding the message bits. Two consecutive message bits are of values 1 and 0. These bits are hidden into two consecutive MBs, say MB_i and MB_{i+1}. In this example, the quantization scale of the first MB is incremented by one to become 9 and the quantization scale of the second MB is decremented by one to become 5. The encoder stores the feature variables of the two MBs with the values shown in Table 1.

Table 1. Example macroblock-level feature variables.

Feature	MB_i	MB_{i+1}
Buffer occupancy	9.903725	0.580925
MB spatial activity	0.741272	9.870142
Quantization scale	9	5

In general, during the encoding process, the encoder stores the feature variables for all MBs, expands them to the second order as described in Equation (4) and computes the model weights as described in Equation (6). The model weights for this particular example and the expanded feature vectors of MB_i and MB_{i+1} are shown in Table 2.

Table 2. Example model weights and expanded feature vectors.

Model weights	Feature vector of MB_i	Feature vector of MB_{i+1}
0.8560	1	1
-219.1380	0.741272	0.580925
-219.5450	9.903725	9.870142
-219.1190	9	5
-0.4380	0.549484	0.337474
0.3030	98.08377	97.4197
-0.3190	81	25
219.4780	19.645	15.45107
186.2050	385.9259	238.7355
-186.3590	14.56229	8.975911
-186.5070	194.5586	152.5042
-185.8970	176.805	77.25534

To decode the message, the decoder computes the feature variables of the MBs from the encoded bitstream, expands them to the second order and uses the model weights to predict the message bits as described in Equation (7). The predicted message bits for the above example are 0.77 and -0.061. With rounding, the predicted bits become 1 and 0 respectively.

It is worth mentioning that one of the reasons for the success of this solution is that the set of feature vectors used at the encoder to generate the model weights is replicated at the decoder. One exception is the macroblock activity feature variable as explained previously. Therefore, the decoder uses the

same model weights and very similar feature vectors to predict the hidden message bits. This results in high prediction accuracy as shall be elaborated upon in the experimental results section.

Lastly, it is worth pointing out that message hiding using this proposed solution can be extended to allow the encoder to hide message bits in selective macroblocks. This is possible if the message extraction process is modified to predict the quantization scale at the decoder. For a given macroblock, if the predicted quantization scale is the same as the one received in the bitstream, then no bits are hidden in that particular macroblock. On the other hand, in typical techniques that use the least significant bits of DCT coefficients to hide message bits, such a selective approach cannot be implemented.

3. Message hiding using Flexible Macroblock Ordering (FMO):

One of the limitations of the quantization scale modulation solution of the previous section is related to the message payload where only one message bit can be hidden per macroblock. This section introduces a second solution that benefits from a higher message bitrate through the use of Flexible Macroblock Ordering (FMO) of the H.264/AVC video coding standard.

In general, a coded picture is divided into one or more slices. Slices are self-contained and can be decoded and displayed independently of other slices. Hence intra prediction of DCT coefficients and coding parameters of a macroblock is restricted to previous macroblocks within the same slice. This feature is important to suppress error propagation within a picture due to the nature of variable length coding. In regular encoding, when FMO is not used, slices contain a sequence of macroblocks in raster scan order. However, FMO allows the encoder to create what is known as slice groups. Each slice group contains one or more slices and macroblocks can be assigned in any order to these slices. The assignment of macroblocks to different groups is signaled by a syntax structure called the 'slice group id'. This syntax structure is available in the picture parameter set header and therefore, can be

altered on picture basis. Notice that the H.264/AVC standard allows for a maximum of 8 slice groups per picture [19].

The idea behind the use of FMO in H.264/AVC is to spread the errors caused by burst packet losses to a larger portion of the picture. As such error concealment becomes easier and more effective. There are a number of predefined slice group types in H.264/AVC that are designed for that purpose. Examples include interleaved slice groups, dispersed slice groups, foreground/background slice groups, box-out and wipe slice groups [20]. The H.264/AVC standard also allows for a sixth type for the explicit assignment of macroblocks to slice groups.

Although FMO was devised for enhancing error resiliency and concealment [21], nonetheless it has been used for other purposes as well. For instance [22] proposed the use of FMO to aid video scrambling for privacy protection. FMO has also been used to enhance the efficiency of video transcoding [23].

In this work we make use of the explicit assignment of macroblocks to slice groups to hide messages in the video stream. Since macroblocks can be arbitrary assigned to slice groups, we propose to use the slice group id of individual macroblocks as an indication of message bits. Assume for instance that 2 slice groups are used, the allocation of a macroblock to slice group 0 indicates a message bit of 0 and the allocation of macroblock to slice group 1 indicates a message bit of 1. Hence one message bit per macroblock can be carried. Furthermore, since the H.264/AVC standard allows for a maximum of 8 slice groups per picture then 2 or 3 message bits can be carried per macroblock as elaborated in Table 3.

Table 3. Number of slice groups versus number of hidden message bits per macroblock.

Number of slice groups	Potential message bits / MB	Message bits / MB
2	0,1	1
4	00,01,10,11	2
8	000,001,010,011, 100,101,110,111	3

Clearly, one can think of other arrangements for assigning message bits to macroblocks. The arrangement given in Table 3 is one straightforward example. Other examples might use 8 slice groups yet use a subset of them for data hiding. For instance, one can use slice groups 2 and 5 to indicate a message bit of 0 and slice group 3 and 7 to indicate a message bit of 1. In general, what can be varied is the size of the subset of slice groups that are used to hide information, the message bit values hidden in these slices groups and the order in which message bits are assigned to slice groups. All of these permutations can even be altered per frame. Such scenarios indicate that the permutations for message hiding using this approach are very large. In this work however, we only consider the straightforward scenarios given in Table 3.

In general to hide a message into the H.264/AVC bit stream, the message is first read into chunks of n bits, were n is either 1,2 or 3 according to the values in Table 3. If m macroblocks are coded per picture, then mxn message bits can be used to allocate the macroblocks to slice groups. The process of message hiding is illustrated in Figure 3.

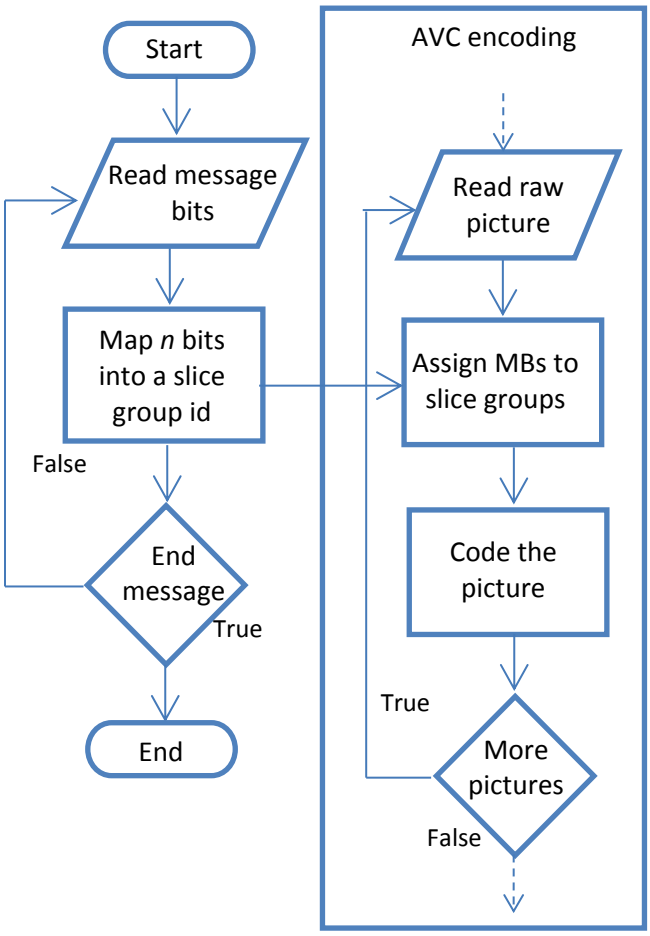


Figure 3. Message hiding using FMO.

To extract the message bits, each time a picture is decoded, the macroblock to slice group mapping syntax structure is used to read *mxn* message bits and append them to the extracted message. The process of message extracting is illustrated in Figure 4.

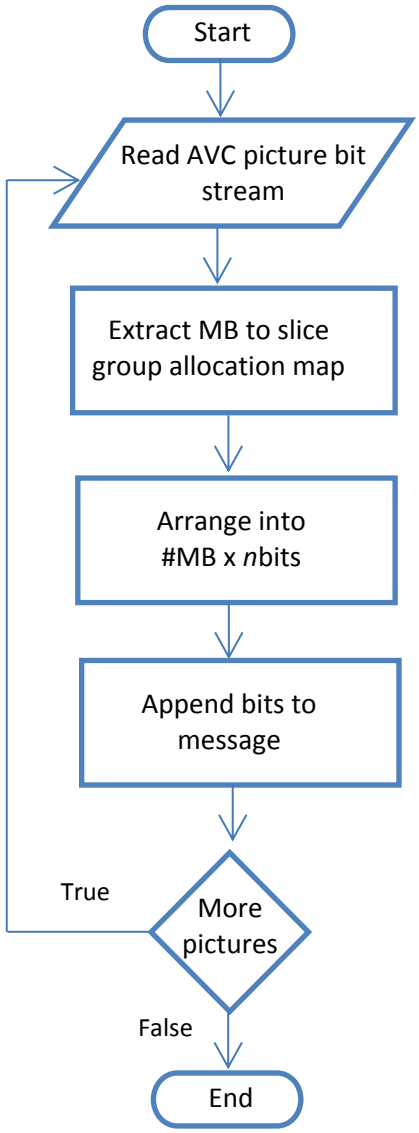


Figure 4. Message extracting using FMO.

3.1 Advantages and disadvantages of the proposed FMO solution

The proposed approach has a number of advantages. It is simple and it is fully complaint with the H.264/AVC syntax using the baseline or the extended AVC profiles. Another advantage is that message hiding works for both coded and skipped macroblocks. The proposed solution also works independent of picture type being I (intra) ,P (predicted) or B (bi-directionally predicted).

In terms of message hiding capacity, if 3 message bits are hidden per macroblock, a message payload of 35.64 Kbits/s is achieved at a 352x288, 30Hz video resolution. Likewise a message payload of

121.5 Kbits/s is achieved at a 720×480, 30Hz video resolution. Lastly, using FMO as means of message hiding does not violate its original purpose, thus, using FMO increases error resiliency and enhances error concealment although this is outside the scope of this paper.

On the other hand, it is well known that the use of FMO increases the bitrate of coded video. In the arbitrary assignment of macroblock to one of 8 slice groups, H.264/AVC adds a syntax element of 3 bits/macroblock to the picture header. However, the bitrate overhead will increase beyond the syntax elements as the FMO interferes with the intra prediction across macroblock boundaries. The macroblocks are no longer coded in raster scan order hence the prediction of DCT coefficients and coding parameters will be less efficient. If Constant Bit Rate (CBR) coding is used then this will affect the video quality and if Variable Bit Rate (VBR) is used then the bitrate is affected. Clearly the extent to which the coded video is affected is content specific as shall be elaborated upon in the experimental results section.

4. Experimental results

This section reports the experimental results of the proposed message hiding solutions and compares them to existing work reported in [8] and [9]. All the messages in the following sections are generated randomly using a uniform distribution of ones and zeros.

4.1 Quantization scale modulation experimental results:

We evaluate the quantization scale message hiding solution using the following criteria:

1. Message prediction accuracy.
2. Message hiding payload which can be measured in Kilobits per second (Kbit/s).
3. The excessive bitrate as a result of message hiding in Kbit/s. This is computed as the difference between the bit rates of the video carrying the message and the original video.
4. Lastly, the drop in PSNR measured in dB.

We compare our proposed solution to a similar work that uses the quantization scale for message hiding as reported in [9]. For a fair comparison, we use similar test sequences and test conditions reported in [9] which are summarized in Table 4. All sequences are compressed using frame-based MPEG-2 encoder at a bitrate of 1.5Mbit/s with a Group of Picture (GoP) structure of N=15 and M=3 (total of 15 pictures per GoP with 2 B-frames between reference frames).

Table 4. Video test sequences

Sequence ID	Sequence Name	#MBs /frame	Frames /sec
V1	Coastguard	396	30
V2	Container	396	30
V3	Flowergarden	330	30
V4	Foreman	396	30
V5	Hall monitor	396	30
V6	Mobile	396	30

The prediction accuracy is computed by decoding a video sequence, extracting macroblock-based features and arranging them into feature matrix. As explained in Section 2.3, the feature matrix is expanded to the second order and multiplied by the 12 model weights to generate the predicted message. To check the accuracy of the predicted message, we compare it bitwise with the original message and report the prediction accuracy. The results are shown in Table 5. The table also shows the results of message prediction without the use of reduced model polynomial expansion as a reference (referred to as a first order model in the table).

Table 5. Accuracy of message prediction.

	Poly. Expansion	
Sequence name	1st order	2nd order
Coastguard	93.8%	98.2%
Container	98.2%	99.96%
Flowergarden	78%	91.6%
Foreman	92%	97.4%
Hall monitor	96.7%	99.8%
Mobile	74.2%	88%
Average	88.7	95.83

It is shown in the table that the message prediction accuracy using second order regression is more accurate than the first order regression. This indicates a nonlinear relationship between the macroblock-based feature variables and the message bits. It is also shown in the table, that for 4 out of 6 sequences, the message prediction accuracy is very high. This means that the proposed solution can be used to hide and extract messages with high accuracy. On the other hand, the prediction accuracy for both the Flower-garden and the Mobile sequences is 91.6% and 87.8% respectively. What is common about these two sequences is the high spatial variance of their images. Recall that in Section 2.1, it was mentioned that the spatial activity is used as a feature variable. It was also mentioned that the encoder uses the spatial activity as a parameter in determining the quantization scale. However, the encoder computes the spatial activity of the original non-compressed macroblocks. For the message extraction on the other hand, the original images are not available, thus the spatial activities of the reconstructed macroblocks are used instead. Further investigation revealed that the Mean Root Square Error (RMSE) between the spatial variance of the original and the reconstructed macroblocks of the above sequences are noticeably higher for the Flower-garden and Mobile sequences. The RMSE values are reported in Table 6.

Table 6. RMSE between the spatial activities of original and reconstructed macroblocks.

Sequence	RMSE
Coastguard	55.1
Container	32.0
Flowergarden	212.8
Foreman	31.2
Hall monitor	33.1
Mobile	178.8

Regardless of the test sequence used, one can conclude that at the time of message hiding, the message prediction accuracy can be assessed by simulating the message extraction procedure. Consequently, if the prediction accuracy is intolerable, then a different video can be used. Clearly the accuracy in message tolerance is message dependent. Text message for instance, are more tolerable than numeric data.

In the following experiment we compare the proposed solution against that reported in [9]. Different configurations are reported in [9], we compare against the case that has the maximum message payload capacity. For fairness, in the proposed work, the message prediction inaccuracy is deduced from the message payload prior to reporting. For instance if the message prediction accuracy in the Coastguard sequence is 98.2% then the message payload is reported with 1.8% less bits. Also recall that if the original macroblock quantization scale was the minimum or the maximum allowed value then the corresponding macroblock will not be used for hiding a bit as explained in Section 2. The comparison results are shown in Table 7.

Table 7. Comparison with existing work in terms of payload, overhead and distortion.

Sequence name	Reviewed [9] Payload Kbit/s	Bitrate overhead Kbit/s	Average distortion [dB]
Coastguard	2.70	284.02	0
Container	0.55	83.08	0
Flowergarden	3.57	502.32	0
Foreman	2.27	206.26	0
Hall monitor	1.23	109.49	0
Mobile	4.36	464.30	0
Average	2.45	274.91	0.00

(a)

Sequence name	Proposed solution Payload Kbit/s	Bitrate overhead Kbit/s	Average distortion [dB]
Coastguard	11.67	0	0.22
Container	11.29	0	0.82
Flowergarden	9.01	0	0.09
Foreman	11.52	0	0.34
Hall monitor	11.81	0	0.55
Mobile	10.34	0	0.08
Average	10.94	0.00	0.35

(b)

While the work in [9] reported no quality distortions, the proposed work reports a constant bit rate at the expense of slight quality degradation. The average drop in PSNR for all sequence is 0.35 dB. On the other hand, the reviewed work reports an increase in bitrate as a result of message hiding. The average increase is around 275 Kbit/s.

Moreover, it is shown that the average message payload or the amount of message bits that can be hidden using the proposed solution is 10.94 Kbit/s. Whereas, in the reviewed solution the message payload is on average 2.45 Kbit/s.

4.2 FMO experimental results:

Unlike the proposed quantization scale modulation solution, there is no message prediction in the FMO solution, hence we evaluate the FMO message hiding solution using the following criteria:

1. Message hiding payload which can be measured in Kilobits per second (Kbit/s).
2. The excessive bitrate as a result of message hiding in Kbit/s
3. Lastly, the drop in PSNR measured in dB.

To be able to compare the FMO message hiding solution with the proposed quantization scale message hiding approach, we use the same video sequences and coding parameters as reported in Table 4 above. The only difference is that H.264/AVC is used instead of MPEG-2 as FMO is not supported in the latter.

In the first set of experiments we use CBR coding at 1.5Mbit/s and observe the drop in PSNR as a result of message hiding using FMO. The results are reported for three cases. The first case uses 2 slice groups per frame hence the maximum bits to hide per macroblock is 1. The second case uses 4 slice groups per frame hence the maximum bits to hide per macroblock is 2. Lastly, the third case uses 8 slice groups per frame and therefore the maximum bits to hide per macroblock is 3.

Table 8. Message hiding results using the proposed FMO with CBR coding.

	FMO (2 Slice Groups)	
Sequence name	Payload Kbit/s	Average distortion [dB]
Coastguard	11.88	0.26
Container	11.88	0.06
Flowergarden	9.9	0.23
Foreman	11.88	0.23
Hall monitor	11.88	0.1
Mobile	11.88	0.22
Average	11.55	0.18

(a)

	FMO (4 Slice Groups)	
Sequence name	Payload Kbit/s	Average distortion [dB]
Coastguard	23.76	0.41
Container	23.76	0.13
Flowergarden	19.8	0.37
Foreman	23.76	0.41
Hall monitor	23.76	0.14
Mobile	23.76	0.43
Average	23.10	0.32

(b)

	FMO (8 Slice Groups)	
Sequence name	Payload Kbit/s	Average distortion [dB]
Coastguard	35.64	0.58
Container	35.64	0.16
Flowergarden	29.7	0.53
Foreman	35.64	0.59
Hall monitor	35.64	0.2
Mobile	35.64	0.61
Average	34.65	0.45

(c)

The results in Table 8 can be compared with those in Table 7. It is clear that the number of bits hidden per macroblock is constant as explained previously. The Flower-garden sequence has 330 macroblock/frame and therefore lower message payload. It is interesting to observe that the use of 4 slices groups, results in an average distortion of 0.32 dB. This is similar to the distortion reported for Table 7 for the quantization scale message hiding approach. Yet, with the FMO approach, the message payload is on average twice as much. It is worth mentioning that a PSNR difference lower than 0.5dB is visually negligible, therefore is it worth noting that all of the distortions for the case of 4 slice groups result in a negligible distortion.

To assess both the excessive bitrate and quality degradations caused by the FMO message hiding solution, we examine the rate-distortion curves for the above test sequences. The results for the case of 2 slice groups are shown in Figure 5.

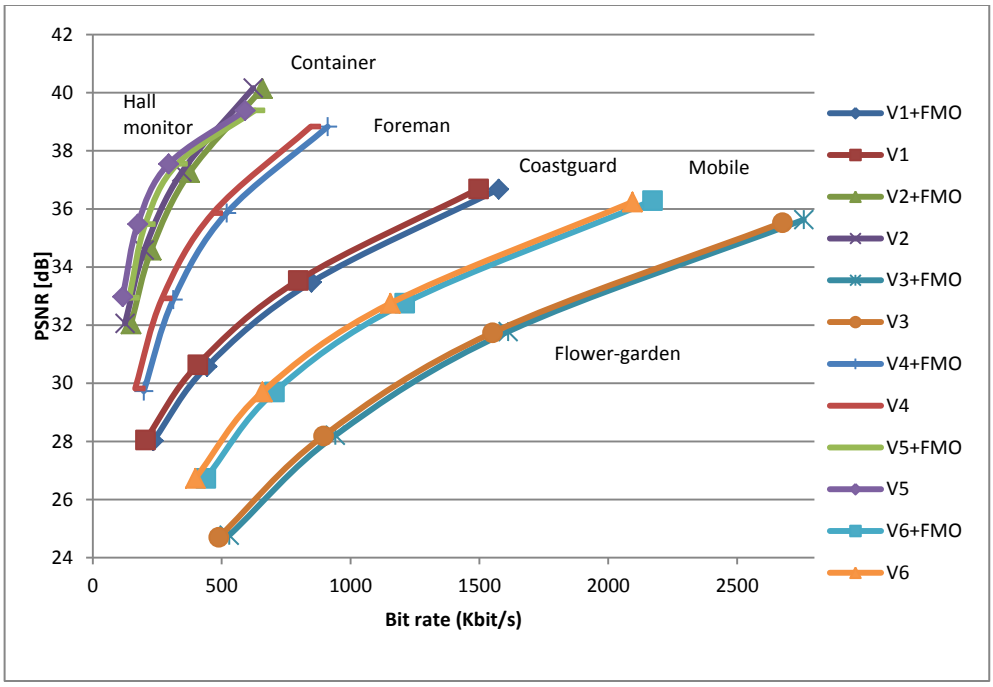


Figure 5. Rate-distortion curves for message hiding using 2 slice groups.

A closer look at the generated bit rates and video quality reveals that the average drop in PSNR in all of the above cases is insignificant. In fact the largest drop for all of the above cases was less than 0.2 dB. On the other hand, the average increment in bit rate was as follows. Flower-garden: 5.13%, Mobile: 6.4%, Coastguard: 8.9%, Foreman: 14%, Container: 11.6%, Hall monitor: 14%. Therefore, it is clear that for video sequences with high spatial variance like Mobile and Flower-garden, the increase in bit rate is lower. The situation is reversed for sequences with low spatial variance like Hall monitor and Container. This is so because with the introduction of arbitrary macroblock ordering, the intra prediction across macroblock boundaries is expected be less efficient. Therefore, sequences that benefits the most out of intra prediction across macroblock boundaries are expected to generate higher bitrates in the case of message hiding.

We further compare the proposed FMO message hiding solution with the work reported in [8]. As mentioned in the introduction, the reviewed work hides the message bits in the MVs. In comparison to hiding message bits using the quantization scale, interframe-coded macroblocks can have more than one MV, therefore, the payload of the embedded message is expected to be higher. However,

changing the values of MVs to embed message bits reduces the quantity of the compressed video. The message bits in this case can be hidden in the x and y components in a subset of the MVs. For a fair comparison we use the same video sequences and coding parameters as in [8]. The following sequences are used at frame rate of 25Hz and coded using VBR coding: Car-phone (176x144 pixels, 270 frames, average bitrate of 424Kbit/s), Foreman (352x288 pixels, 135 frames, average bitrate of 1150Kbit/s), Football (352x240 pixels, 117 frames, average bitrate of 1600 Kbit/s), Coastguard (352x288 pixels, 270 frames, average bitrate of 1660Kbit/s) , Mobile (352x288 pixels, 270 frames, average bitrate of 1830 Kbit/s) and Flower-Garden (352x288 pixels, 250 frames, average bitrate of 1670 Kbit/s) . The GoP structure used is N=9 and M=3 (total of 9 pictures per GoP with 2 B-frames between reference frames). Full motion estimation range is used in both the reviewed and the proposed work. The results of the reviewed work are reported in Table 9.

Table 9. Message hiding result of reviewed work [8]

Sequence name	Reviewed work [8]		
	Payload Kbit/s	Distortion [dB]	Bitrate overhead %
Car-phone	3.29	0.40	3.96
Foreman	8.44	0.22	12.58
Football	16.17	0.29	6.09
Coastguard	13.35	0.27	7.49
Mobile	14.70	0.60	18.06
Flowergarden	15.70	0.55	13.19
Average	11.94	0.39	10.23

The corresponding message payloads, distortions and bitrate overhead of the proposed work are reported in Table 10.

Table 10. Results of FMO message hiding using VBR coding.

Sequence	Proposed FMO (2 slice groups)		
	Payload Kbit/s	Distortion [dB]	overhead %
Car-phone	2.48	0	6.39
Foreman	9.9	0.1	7.95
Football	8.25	0	3.96
Coastguard	9.9	0.2	4.54
Mobile	9.9	0.1	3.67
Flowergarden	9.9	0	3.00
Average	8.39	0.07	4.92

(a)

Sequence	Proposed FMO (4 slice groups)		
	Payload Kbit/s	Distortion [dB]	overhead %
Car-phone	4.95	0	11.51
Foreman	19.8	0.1	14.17
Football	16.5	0	6.86
Coastguard	19.8	0.2	8.21
Mobile	19.8	0.1	6.85
Flowergarden	19.8	0	5.95
Average	16.78	0.07	8.93

(b)

Sequence	Proposed FMO (8 slice groups)		
	Payload Kbit/s	Distortion [dB]	overhead %
Car-phone	7.43	0	16.41
Foreman	29.7	0.1	19.22
Football	24.75	0	9.37
Coastguard	29.7	0.2	11.74
Mobile	29.7	0	9.68
Flowergarden	29.7	0.1	8.80
Average	25.16	0.07	12.54

(c)

The results show that the PSNR drop in the proposed solution is insignificant and is equal to zero in 3 out of the 6 test sequences. It is also shown in the table that with the use of 4 slice groups the results are slightly better than the reviewed work. However, one advantage of the proposed work is the ability of increasing the message payload up to 25.16 Kbit/s as shown in the table. In that case, the percentage bitrate overhead is 12.54% as

opposed to the reviewed work where 10.23% is reported. However, the average payload in the reviewed work is 11.94 Kbit/s which is about half the payload of the proposed work when 8 slice groups are used.

5. Conclusion.

The paper proposed two novel approaches to message hiding. In the first approach, the quantization scale of a CBR video is either incremented or decremented according to the underlying message bit. A second order multivariate regression is used to associate macroblock-level features with the hidden message bit. The decoder makes use of this regression model to predict the message bits. It was shown that high prediction accuracy can be achieved. However, the message payload is restricted to one bit per macroblock. The second approach proposed in the paper works for both CBR and VBR coding and achieves a message payload of 3 bits per macroblock. The FMO was used to allocate macroblocks to slice groups according to the content of the message. Comparisons with existing work revealed the effectiveness of the proposed solutions in terms of message payload, video distortion and excessive overhead. Future work includes examining the robustness of the proposed work against channel bit errors, packet losses and existing digital video steganalysis methods.

References.

[1] M. Carli, M. Farais, E. Drelie Gelasca, R. Tedesco, A. Neri, "Quality assessment using data hiding on perceptually important areas," IEEE International Conference on Image Processing, ICIP 2005, pp. III- 1200-3, September 2005.

[2] S. Kapotas, A. Skodras, "A new data hiding scheme for scene change detection in H.264 encoded video sequences," IEEE International Conference on Multimedia and Expo ICME 2008, pp.277-280, June 2008.

[3] A. Yilmaz, A. Aydin, "Error detection and concealment for video transmission using information hiding," Signal Processing: Image Communication, 23(4), pp. 298-312, April 2008

[4] Y. Li, H.-X. Chen, Y. Zhao, "A new method of data hiding based on H.264 encoded video sequences," IEEE International Conference on Signal Processing, ICSP 2010, pp.1833-1836, October 2010.

[5] K. Nakajima, K. Tanaka, T. Matsuoka, and Y. Nakajima, "Rewritable data embedding on MPEG coded data domain," IEEE International Conference on Multimedia and Expo, ICME 2005, pp. 682–685, July 2005.

[6] D.-Y. Fang, L.-W. Chang, "Data hiding for digital video with phase of motion vector," IEEE International Symposium on Circuits and Systems, ISCAS 2006, September 2006.

[7] C. Xu, X. Ping, and T. Zhang, "Steganography in compressed video stream," International Conference on Innovative Computing, Information and Control, ICICIC'06, vol. II, pp. 803–806, 2006.

[8] H.A. Aly, "Data Hiding in Motion Vectors of Compressed Video Based on Their Associated Prediction Error," IEEE Transactions on Information Forensics and Security,6(1), pp.14-18, March 2011.

[9] K. Wong, K. Tanaka, K. Takagi and Y. Nakajima, "Complete Video Quality-Preserving Data Hiding," IEEE Transactions on circuits and systems for video technology, 19(10), October 2009.

[10] Y. Hu, C. Zhang and Y. Su, "Information Hiding Based on Intra Prediction Modes H.264/AVC," IEEE International Conference on Multimedia and Expo, ICME 2007, pp.1231-1234, July 2007.

[11] G. Yang, J. Li, Y. He and Z. Kang, "An information hiding algorithm based on intra-prediction modes and matrix coding for H.264/AVC video stream," International Journal of Electronics and Communications, (65)4, pp. 331-337, April 2011.

[12] K. Solanki, U. Madhow, B.S. Manjunath, S. Chandrasekaran and I. El-Khalil, "`Print and Scan' Resilient Data Hiding in Images," IEEE Transactions on Information Forensics and Security, 1(4), pp.464-478, December 2006

[13] X.-P. Zhang, K. Li and X. Wang, "A Novel Look-Up Table Design Method for Data Hiding With Reduced Distortion," IEEE Transactions on Circuits and Systems for Video Technology, 8(6), pp.769-776, June 2008

[14] Ma Xiaojing, Li Zhitang, Tu Hao and Zhang Bochao, "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), pp.1320-1330, October 2010

[15] S. Lyu and H. Farid, "Steganalysis using higher-order image statistics," IEEE Transactions on Information Forensics and Security, 1(1), pp. 111- 119, March 2006.

[16] U. Budhia, D. Kundur and T. Zourntos, "Digital Video Steganalysis Exploiting Statistical Visibility in the Temporal Domain," IEEE Transactions on Information Forensics and Security, 1(4), pp.502-516, Dec. 2006

[17] K. Wong and K. Tanaka, "A Data Hiding Method Using Mquant in MPEG Domain", Journal of the Institute of Image Electronics Engineers of Japan, 37(3), pp.256-267, 2008.

[18] K.-A Toh, Q.-L. Tran and D. Srinivasan, "Benchmarking a Reduced Multivariate Polynomial Pattern Classifier," IEEE Transactions on pattern analysis and machine intelligence, 26(6), June 2004.

[19] Recommendation ITU-T H.264 | ISO/IEC 14496-10:2009, "Advanced Video Coding for generic audio-visual services," March 2009.

[20] S. Wenger, M. Horowitz, "Flexible Macroblock Ordering (FMO) 101," available from http://ftp3.itu.ch/av-arch/jvt-site/2002_07_Klagenfurt/JVT-D063.doc, July 2002.

[21] S. Wenger, “H.264/AVC over IP,” IEEE Transactions of Circuits Systems for Video Technology, vol. 13, pp. 645–656, July 2003.

[22] F. Dufaux and T. Ebrahimi, "H.264/AVC video scrambling for privacy protection," IEEE International Conference on Image Processing, ICIP 2008, pp.1688-1691, October 2008.

[23] J. Lievens, P. Lambert, D. Walle, F. Dawoud, J. Barbarien, R. Walle and P. Schelkens, "Compressed-domain motion detection for efficient and error-resilient MPEG-2 to H.264 transcoding", Proc. SPIE 6696, 669609 (2007).

Tamer Shanableh earned his Ph.D. in Electronic Systems Engineering in 2002 from the University of Essex, UK. From 1998 to 2001, he was a senior research officer at the University of Essex, during which, he collaborated with BTexact on inventing video transcoders. He joined Motorola UK Research Labs in 2001. During his affiliation with Motorola, he contributed to establishing a new profile within the ISO/IEC MPEG-4 known as the Error Resilient Simple Scalable Profile. He joined the American University of Sharjah in 2002 and is currently an associate professor of computer science. Dr. Shanableh spent the summers of 2003, 2004, 2006, 2007 and 2008 as a visiting professor at Motorola multimedia Labs. His research interests include digital video processing and pattern recognition.