Data Embedding in Scrambled Video by Rotating Motion Vectors

Afaf Eltayeb[1] and Tamer Shanableh[2*]
[1]Department of Computer Science and Engineering, American University of Sharjah, UAE, g00082016@aus.edu
[2]Department of Computer Science and Engineering, American University of Sharjah, UAE, tshanableh@aus.edu

***Abstract*— Data embedding in videos has several important applications including Digital Rights Management, preserving confidentiality of content, authentication and tampering detection. This paper proposes a novel data embedding solution in scrambled videos by rotating motion vectors of predicted macroblocks. The rotation of motion vectors and the propagation of motion compensation error serve another purpose, which is video scrambling. A compliant decoder uses machine learning to counter-rotate the motion vectors and extract embedded message bits. To achieve this, the decoder uses a sequence-dependent approach to train a classifier to distinguish between macroblocks reconstructed using rotated and un-rotated motion vectors. In the testing phase, motion vectors belonging to a classified macroblock are compared against the reviewed rotated motion vectors and the message bits are extracted. Furthermore, to guarantee accurate classification at the decoder, a constrained encoding approach is proposed in which data embedding is restricted to motion vectors that can be correctly counter-rotated at the decoder. The proposed solution is referred to as Classifying Rotated Vectors or CRVs for short. Experimental results revealed that scrambled videos can be reconstructed correctly without quality loss with a bitrate increase at the encoder of around 6% and an average data embedding rate of 1.68 bits per MB.**

Keywords—Data embedding, Machine learning, Video coding

I. INTRODUCTION

Data embedding in videos has various important applications including Digital Rights Management (DRM), embedding of authentication data and owner identity information. Data embedding is also used for tampering detection and ownership declaration of digital content.

Data embedding can be applied to compressed images and videos directly or to compressed and scrambled/encrypted images and videos. In this research domain, encryption refers to scrambling of visual content or compression of information. It typically does not refer to encryption in the computer security sense. The scrambling or encryption of digital content has several applications such as privacy protection of specific regions in surveillance images and videos, and privacy protection for images and videos stored on the cloud.

Data embedding in the compressed domain without scrambling or encryption has a drawback of increasing the image/video size, degrading their visual quality or both. On the other hand, data embedding in the scrambled or encrypted domain has the advantage of lossless recovering of both embedded data and cover image/video.

Examples of data embedding in compressed video domain include modulating quantization step sizes [1], modulating Motion Vectors (MVs) [2] and modulating coding parameters of image blocks [3].

In this work, on the other hand, we focus on data embedding in the scrambled domain. Such an approach can be applied to both images and videos. For instance, the authors in [4] proposed data embedding in encrypted images where data is hidden by flipping three LSBs of pixels in a given pixel block. A block smoothness measure is used to extract the data and recover the image. The work in [5] implemented a system in which scrambling and data embedding are simultaneous. The DC coefficients, statistical properties of AC coefficient block, and length of zero AC coefficients are utilized for scrambling and data embedding. Additionally, in [6], the authors proposed to reserve a room from images prior to encryption to embed data using a traditional reversible data-hiding algorithm. The hidden data is recoverable and image reconstruction is error-free.

Likewise, data embedding can be implemented in compressed and scrambled/encrypted video. For instance, the work in [7] proposed to embed data in the encrypted H.264 videos to preserve the confidentiality of the video. Data embedding in the encrypted video is performed using codeword substitution, thus, the original video content does not need to be known to the data hider. In [8], a histogram shifting technique is used for data embedding. The receiver can decrypt the video and extract the data without affecting the quality of the video. On the other hand, if the data is not extracted then the video can be decrypted and reconstructed at an acceptable quality.

In the literature, some research work proposed application-oriented data embedding solutions, for example in [9]; video regions of interest are identified and scrambled for privacy protection purposes. This is achieved by encrypting the modes of intra prediction and encrypting the signs of nonzero Discrete Cosine Transform (DCT) coefficients within the area of interest. The work in [10] proposed a solution based on blockchain for digital rights management. The solution provides trusted content protection and content violation traceability where DRM license and usage control can be retrieved form the blockchain. In addition, the work in [11] proposed a confidentiality preserving solution of videos where data is embedded directly in partially encrypted H.264 videos by substituting bin strings of the context-adaptive binary arithmetic coder.

In this work, we extend the existing literature by proposing a machine learning approach to data embedding in the scrambled video domain. In brief, we propose to embed messages by rotating motion vectors of the cover video. This is applied to the bitstream of the video, hence does not affect the motion compensation process of video encoding. At the receiver, the video can be decoded correctly, however the visual content is scrambled and therefore the watching experience is unpleasant, which is the sole objective of video scrambling. To unscramble the video and extract the embedded data, we propose a machine

learning solution that distinguishes between the correctly reconstructed macroblocks and those reconstructed using rotated MVs.

It is known that such solutions require model generation or training. We propose a video-dependent solution where the first part of the video is used for model generation. Such an approach was successfully implemented in speeding up High Efficiency Video Coding (HEVC) transcoding [13]. On the other hand, video-independent training was successfully implemented for Coding Unit (CU) split prediction in HEVC videos [12].

The video-independent approach has the advantage of being able to scramble the whole cover video, yet it has a disadvantage of one-size-fits-all, which is not guaranteed to work with all video contents. The video-dependent approach on the other hand, has the advantage of being specific for the underlying cover video and therefore guaranteed to generate a more suitable model, yet it has the disadvantage of using the first part of the video for model generation. Therefore, the first part of the video will not carry embedded data.

Another difference between the two approaches is that video-independent model generation is implemented off-line and video-dependent model generation is implemented on-line at the decoder's side.

In our work, the decoder receives a motion vector with embedded data, the decoder will then generate 4 candidate motion vectors with all possible rotations. If a macroblock has 2 MVs then 16 combinations are generated. The 4 or 16 motion vectors are then used for motion compensation and 4 or 16 candidate macroblocks are reconstructed. Only one of these macroblocks belongs to the original un-rotated motion vector. At this point, the decoder can use the classification model to decide which out of the 4 or 16 macroblocks is the one belonging to the un-rotated motion vector. As such, the classified motion vector can be compared to the one received in the scrambled video and by doing so, the embedded data is extracted. Once the original, un-rotated motion vectors are classified, the video can be decoded into its unscrambled state.

The rest of this paper is organized as follows. Section II introduces the proposed data embedding solutions. Section III introduces the proposed feature extraction and model generation. Section IV explains the deployment of the trained model for message extraction. Section VI lists the configuration of the classification tools used. Section VII presents detailed experimental results and Section VIII concludes the paper.

II. Proposed Data Embedding

The proposed solutions of this paper are implemented using two video coding standards, namely, Moving Picture Experts Group (MPEG2) and High Efficiency Video Coding (HEVC) video coding standards. In video coding a raw video sequence is compressed into a specific bit stream format by the encoder. The decoder on the other hand decompresses bit streams back into raw video sequences with reduced quality due to quantization. The encoder performs Motion Estimation (ME) between video frames which results in a sequence of Motion Vectors (MVs). ME can be applied between a current frame and previous frames, future frames or both. This results in forward, backward and bi-directional predictions respectively. The ME is followed by Motion Compensation (MC), transform coding, quantization and Variable Length Coding (VLC). While the decode performs variable length decoding, inverse quantization, inverse transformation and motion compensation using the received motion vectors.

In the proposed system, the message bits are hidden by rotating MVs by 0, $\prod/2$, $\prod$ or $3\prod/2$ degrees. With these four angles, 2 message bits can be hidden in one MV. The rotation of a MV is given in (1):

$$\begin{bmatrix} V_x'^{(j)} \\ V_y'^{(j)} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} V_x^{(j)} \\ V_y^{(j)} \end{bmatrix} \quad (1)$$

Where θ is the rotation angle, $V_x^{(j)}$, $V_y^{(j)}$ are the x and y components of the original j^{th} MV, and $V_x'^{(j)}$ and $V_y'^{(j)}$ are the x and y components of the rotated MV. The 4 rotating angles guarantee that the x and y components of MVs can be represented by integer types. Therefore, counter rotation will not result in data loss.

Some MBs are bi-directionally predicted and contain forward and backward motion vectors (FMV and BMV). In such cases, 2 message bits can be embedded in each MV, resulting in 4 message bits per MB. However, in video compression algorithms where MVs are not allowed to exceed image boundaries, only one bit can be embedded per MV belonging to MBs at the image boundaries, the embedding can be in either the horizontal or the vertical component of the MV. This results in a single hidden bit per a predicted MB and 2 hidden bits per a bidirectional MB.

The rotation of the MVs is implemented as a post-process at the video encoder to make sure that the rotated MVs are not used in the motion compensation loop of the encoder. Therefore, by counter-rotating the MVs at the decoder, the video can be reconstructed and displayed correctly.

Clearly, MBs that are intra-coded and MBs with zero MVs are not used for message embedding.

Without counter-rotation of the MVs at the decoder, the video can be correctly decoded without crashing, however, the visual content will be scrambled. The reason is that motion compensation is performed using rotated MVs and therefore, video images are reconstructed incorrectly. Additionally, this error will propagate throughout the video because of the nature of motion compensation.

III. Proposed System Training

The decoder receives rotated MVs based on embedded message bits. For the decoder to be able to counter-rotate the MVs, the decoder needs to classify MVs into rotated/un-rotated. Therefore, we propose a machine learning approach to classify MVs into rotated/ counter-rotated and thereafter extract the embedded message bits.

The system training is sequence-dependent and uses the first 10% of the video sequence to generate the training model. Such an approach has been successfully used in the context of expediting the process of video transcoding and video encoding as reported in [13] and [14].

This implies that the first 10% of the video sequence is not used for message embedding. If scrambling is needed for the

first 10% of the video, then a prearranged MV rotation approach can be used such as rotating all MVs with 180 degrees for instance. Any other reversible video scrambling technique can be used as well for the first part of the video.

The proposed system training is illustrated in the process flow diagram of Fig. 1. Having received the macroblocks (MBs) and their MVs, the training system at the decoder, rotates the MVs using the four angles and uses the rotated MVs to reconstruct four candidate MBs using Motion Compensation (MC). In case a MB has 2 MVs, like forward and backward MVs, then the total number of candidate MBs after MV rotations is 16. This is so as each MV can be rotated using four angles, and the total number of MV combinations for two MVs is 16.

In all cases, Feature Vectors (FVs), are computed from each candidate MB and its rotated MV(s) to form a feature matrix. Since the first 10% of the video is used for model generation, the un-rotated MVs are known and are therefore used as the ground truth in model generation. A binary classifier then learns whether a candidate MB corresponds to the true un-rotated MV or not. Details of feature extraction are given later in this section and details of the classification tools used and their configurations are given in Section VI.

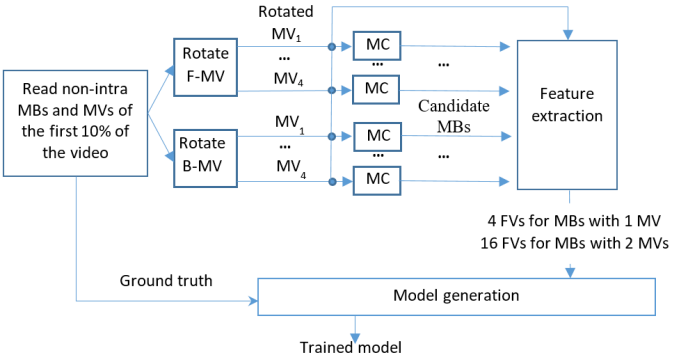


Fig. 1. Process flow diagram of proposed sequence-dependent system training.

For message extraction, as will be explained in the next section, the angle between the received MV and the MV classified as un-rotated is used to extract the message bits.

The features used in the proposed machine learning solution are listed in Table I.

Feature variables 1-4 are related to differences between the phases of the candidate MV and the surrounding ones. A lower difference indicates that the candidate MV is likely to be the true un-rotated MV. Feature 5 is the count of clip violation after adding the prediction error to one of the motion compensation locations using different candidate MVs. The violations are summations outside the pixel range of [0,255]. Lower number of clip violations indicate that the candidate MV is likely to be the true un-rotated MV. Feature 6 is an indication of edge strength after motion compensation using different candidate MVs. The true un-rotated MV is expected to reconstruct the MB correctly with clear edges. Lastly, features 7-11 are statistics from the reconstructed MB using different candidate MVs. Adding the prediction error to different motion compensation locations is expected to generate reconstructed MBs with different statistical features.

Each of the above features are calculated per candidate MV and/or resultant candidate MB. Within a set of candidate MVs and candidate MBs belonging to the same true MB, the feature values are sorted and ranked from 1-4 in P-MBs and 1-16 in B-MBs. If features form different candidate MVs/MBs have the same value then they will be assigned to the same rank.

IV. Proposed Message Extraction

Once the model is generated at the decoder, the system can switch to the testing mode in which MVs are classified as rotated or not. This process is illustrated in the flow diagram of Fig. 2. The feature extraction part is identical to that of the system's training counterpart of Fig. 1. For MBs with a single MV, 4 candidate MBs are generated using 4 rotated MVs. Likewise, for MBs with 2 MVs, 16 candidate MBs are generated using 16 rotated MVs (i.e. 4 rotations for the forward MB x 4 rotations for the backward MV).

Features are extracted from each candidate MB and its corresponding rotated MV(s). Each of the 4 or 16 feature vectors per MB is classified using the trained binary classifier. In an ideal situation, only one out of all candidate MBs will be classified as positive. The corresponding MV is then compared against the decoded MV and the difference in the angle reveals the embedded message bits. If a MB has 2 MVs then the message bits are extracted twice. The extracted bits for the j^{th} MB are computed using (2):

TABLE I
List of Feature Variables Representing Candidate MBs

Feature ID	Description
1	The absolute difference between the phase of the candidate forward MV (FMV) and the phase of the co-located MV of the reference frame.
2	The absolute difference between the phase of the candidate MV and the phase of the top MV of the same frame.
3	The absolute difference between the phase of the candidate MV and the phase of the left MV of the same frame.
4	Same as feature ID 1 above but applied to the backward MV (BMV).
5	Number of pixel clipping violations after reconstructing a candidate MB. The pixel range used for clipping is 0 to 255.
6	Sum of absolute edge values of a candidate MB using edge detection.
7	Pixel entropy of a candidate MB.
8	Pixel mean of a candidate MB.
9	Pixel variance of a candidate MB.
10	The sum of absolute values of the prediction error obtained by subtracting a candidate forward MB (FMB) from the co-located MB in the reference frame.
11	Same as feature ID 10 above but applied to the backward MB (BMB).

$$MB^{(j)} msg\ bits = \begin{cases} 00, \left|\tan^{-1}(V_y^{(j)}/V_x^{(j)}) - \tan^{-1}(V'^{(j)}_y/V'^{(j)}_x)\right| = 0 \\ 01, \left|\tan^{-1}(V_y^{(j)}/V_x^{(j)}) - \tan^{-1}(V'^{(j)}_y/V'^{(j)}_x)\right| = \Pi/2 \\ 10, \left|\tan^{-1}(V_y^{(j)}/V_x^{(j)}) - \tan^{-1}(V'^{(j)}_y/V'^{(j)}_x)\right| = \Pi \\ 11, \left|\tan^{-1}(V_y^{(j)}/V_x^{(j)}) - \tan^{-1}(V'^{(j)}_y/V'^{(j)}_x)\right| = 3\Pi/2 \end{cases} \quad (2)$$

Where $V_x^{(j)}$ and $V'^{(j)}_x$ denote the x component of the received and classified MVs of the j^{th} MB respectively.

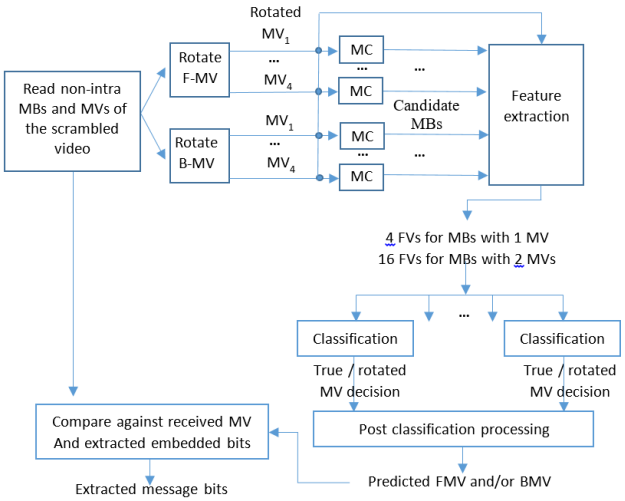


Fig. 2. Process flow diagram of proposed message extraction.

However, there are few situations where the classification system will result in Type I and Type II errors.

In particular, a Type I error occurs If more than one candidate MB are positively classified and a Type II error occurs if all candidate MBs are negatively classified. Formally, let $\mathbf{c}^{(j)}$ to denote the list of classification results (class 1 for positive instances and class 0 otherwise) for the j^{th} MB:

$$\mathbf{c}^{(j)} = \left[c_1^{(j)}, \dots, c_n^{(j)}\right] \tag{3}$$

Where n is the number of candidate MBs, which is either 4 or 16. And $c_i^{(j)}$ denotes the binary classification result of the i^{th} candidate of the j^{th} MB.

Two types of classification error are defined:

$$Err\ type = \begin{cases} I, \sum_{i=1}^{n} c_i^{(j)} > 1 \\ II, \sum_{i=1}^{n} c_i^{(j)} = 0 \end{cases} \tag{4}$$

Type I and II errors can be dealt with using a post classification approach as illustrated in Fig. 2. For MBs with either error type, the Sum of Absolute Differences (SADs) between the boundaries of a candidate MB and its top/left MBs is computed. The candidate MB with the least boundary SAD is identified as the true MB and its MV is identified as the un-rotated MV. The SAD between a MB and its top/left previously decoded MBs is illustrated in Fig. 3.

Using this post-processing approach, the percentage of Type I and Type II errors can be reduced and thus increasing the overall classification accuracy of the system. In the experimental results section, we report the percentage of MBs with Type I and Type II errors, we also report the percentage of MBs with Type I and Type II errors that are correctly classified after applying the proposed post-processing approach.

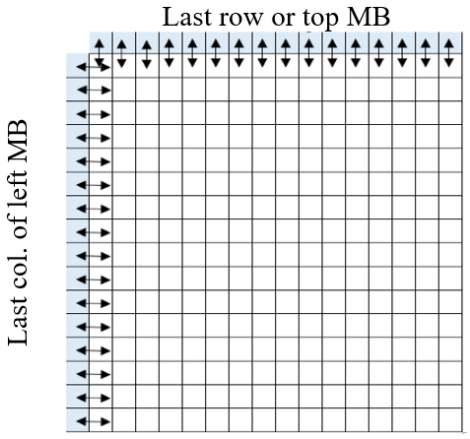


Fig. 3. SAD between boundaries of a candidate MB and its top/left decoded MBs.

There is a third type of error which is due to classification error. In such a case, only one candidate MB is classified positively, however it is not the correct one. No further processing is used in such a case.

V. Combined Encoder-Decoder Solution

This solution complements the decoder-side operations, hence the title "combined encoder-decoder solution". The core idea of this solution is to identify the MBs whose MVs will not be correctly counter-rotated at the decoder due to misclassification. Once identified, the encoder avoids embedding message bits in these MVs. One approach to achieve this is to force the use of a zero MV or use intra coding. This is the case as the decoder is not expected to extract message bits form zero MVs or intra coded MBs. As such, only the MVs that can be correctly counter-rotated at the decoder are used for data embedding. Once received by the decoder, the later performs the proposed model generating and classification as described in the previous section.

This combined encoder-decoder solution can be achieved by replicating the model generation at the encoder side using the same set of frames that the decoder uses for model generation. Thus, the same classification model can be made available at both the encoder and the decoder without the need for communicating the model's parameters.

Once the model is generated at the encoder, the following steps are followed: for each inter MB, the MV is computed and rotated according to the message bits to be embedded. The encoder then replicates the process of the decoder, that is, 4 or 16 candidate MBs and rotated MVs are computed and used for computing features vectors. The feature vectors are then classified using the encoder's side model as illustrated in Fig. 2. If correctly classified, then data embedding and MV rotation take place. Otherwise, the current MB is not used for data embedding. Again, this can be achieved by imposing the use of zero MVs or intra coding. This arrangement is possible at the encoder as the ground truth is known, which is the true MV(s) of the MB being examined. This process is illustrated in the diagrams of Fig. 4.

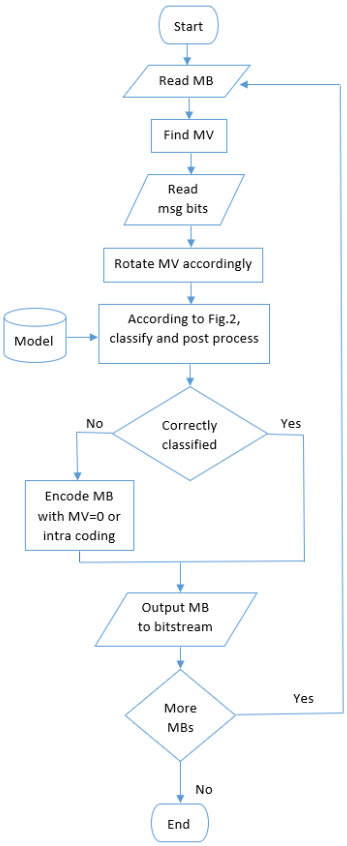


Fig. 4. Data embedding in selective MBs at the encoder's side.

This proposed solution has advantages and disadvantages. The advantages manifest themselves at the decoder side where all received MVs can be classified correctly and therefore counter-rotated correctly. This results in correct message extraction and correct and accurate decoding without any error propagation.

The disadvantages are caused by interfering with the MB decision type at the encoder's side and therefore slightly increasing the bitrate of the generated video. The other disadvantage is the reduced message embedding rate as the percentage of MBs without MVs will increase. However, since the classification accuracy of the proposed decoder-side solution is above 96% (as will be illustrated in the experimental results section), only a small percentage of inter MBs will be affected at the encoder. Therefore, the impact on bitrate and message embedding rates is not significant. These observations are quantified in the experimental results section.

VI. Classification Tools

In this work, we experiment with 3 classification tools, namely; Random Forests [15], SVM and Polynomial classifiers [16].

In the Random Forest classifier, the number of trees grown is 128. We also experiment with feature variable selection in combination with Random Forests. In this approach, a large set of trees are generated against the label of a candidate MB. Each tree is trained on a subset of feature variables from Table I. The usage statistics of each variable are used to find an informative subset of feature variables. In particular, if a feature variable is selected repeatedly as best split, it is considered as a good candidate to keep. More information about this algorithm can be found in [17].

The importance of each of these feature variables in predicting the correct classification of a test feature vector from an out-of-bag data is computed and used to select the feature variables whose raw importance score makes up 90% of the total importance score. The out-of-bag data represent the set of feature vectors that were left out during the training process of a given tree in the random forest.

For the SVM, we train the classifier with an Radial Basis Function RBF kernel of the following form [18]:

$$Kernal(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|} \quad (5)$$

Where x_i, x_j are the i^{th} and j^{th} feature vectors.

For Polynomial classifiers, we implement the work reported in [16] to expand the feature vectors into higher dimensionality using polynomial expansion. The classification decision is then based on the following formula that combines the model weights with the expanded feature variables:

$$f(\boldsymbol{w},\boldsymbol{x}) = \alpha_0 + \sum_{k=1}^{r}\sum_{j=1}^{l} w_{kj}x_j^k + \sum_{j=1}^{r} w_{rl+j}(x_1 + x_2 + \cdots + x_l)^j + \sum_{j=2}^{r}(\boldsymbol{w}_j^T \boldsymbol{x})(x_1 + x_2 + \cdots + x_l)^{j-1}, l, r \geq 2 \quad (6)$$

Where r is the order of the polynomial expansion, $\boldsymbol{w}$ is the vector of polynomial weights to be estimated, $\boldsymbol{x}$ is the feature vector containing l inputs. The dimensionality of the expanded vector is equal to $1+r+l(2r-1)$. The polynomial weights are estimated using least-squares error minimization. In this work, we use a fifth order expansion which is determined experimentally.

VII. Experimental Results

In the following experiments, we use well-known test video sequences with various resolutions and spatio-temporal activities. The sequences are listed in Table II.

TABLE II
HEVC Video Test Sequences and Their Resolutions

	Sequence	WidthxHeight	f/s
1	RaceHorses	416x240	30
2	BlowingBubbles	416x240	50
3	BasketballPass	416x240	50
4	RaceHorses	832x480	30
5	BQMall	832x480	60
6	BasketballDrill	832x480	50
7	ParkScene	1280x768	24
8	Kimono1	1280x768	24
9	Cactus	1280x768	50

The proposed solution is applicable to video coders with block-based motion estimation and compensation such as MPEG2, AVC/H.264 and HEVC. As a proof of concept and for ease of software implementation, the MPEG-2 software codec

is modified to implement the proposed data embedding solution. The sequences are encoded using the following sequence of frame types: IBPBP…, we also repeat the results using an IPPP… sequence. Only the first frame is intra-coded and the rest of the frames are either P or B frames. Additionally, each sequence is compressed with three quantization scales: 5, 15 and 25 out of 31.

We test our solution on three types of MBs, namely; Predicted MBs in P-frames (P-MBs), mono-directional MBs in B-frames, which are predicted using either forward or backward prediction (mono-B-MBs) and bidirectional MBs in B-frames, which are predicted using both forward and backward prediction at the same time (bi-B-MBs).

For training, the first 10% of each video sequence is used to generate the classification model. Therefore, the first 10% of the sequences are not used for data embedding as explained previously. In concept, the training can be repeated periodically or whenever a scene change is detected. A similar training approach was implemented in [13] and [14]. We experiment with four classifiers including Random Forests, Random Forests with feature selection, SVM and a polynomial classifier. In all cases, the messages to embed are generated using a uniform binary random number generator.

To make the reported experimental results readable, we present the averages over all video test sequences in the experiments to follow.

We start by presenting the classification accuracy of data extraction at the decoder's side. As mentioned previously, the decoder generates 4 or 16 candidate MBs and extracts features from each of them. The feature vectors are classified to find out the true un-rotated MVs. Once done, the classified MVs are compared against what is received in the video's bit stream and the embedded message bits are extracted.

Table III presents the classification accuracy and f1-scores of data extraction using RFs with a frame sequence of IBP. The f1-score is the harmonic mean of the precision and recall

Listed in the table are the percentages of Type I, Type II and classification errors. Again, as mentioned previously, Type I error refers to the case where more than one candidate MB were classified as positive. Type II error refers to the case where none of the candidate MBs was classified as positive. Lastly, the classification error refers to the case where one candidate MB was wrongly classified as positive. This error cannot be identified at the decoder and therefore cannot be post-processed.

The Type I and Type II errors are post-processed using SAD of the MB edges as proposed in Section IV above. The percentages of corrected Type I and Type II errors are presented in Table III as well. The post processing is not perfect; however, it assists in increasing the overall classification accuracy.

All of the average classification accuracies are above 96%. There are two additional conclusions that can be drawn from the results of this table. First, the number of candidate MBs is either 4 (in P-MBs and mono-B-MBs) or 16 (in bi-B-MBs). Yet, despite the increase in the number of candidate MBs, the classification accuracy of the later MBs is around 97.5%. The average classification rates over all quantization scales are 96.06%, 96.83% and 97.5% for the P, mono-B and bi-B MBs respectively. The overall classification accuracy using Random Forests for all cases is 96.8%.

TABLE III
CLASSIFICATION ACCURACY OF MESSAGE EXTRACTION AT DECODER USING RANDOM FORESTS AVERAGED OVER 9 SEQUENCES USING A QUANTIZATION SCALE OF (A) 5 (B) 15 AND (C) 25

	P	mono-B	bi-B
Type I err (%)	1.74	1.01	1.01
Type II err (%)	1.87	2.08	1.01
Class. err (%)	2.54	2.23	2.02
Type I err corrected (%)	68.06	81.67	95.09
Type II err corrected (%)	54.35	48.05	93.22
Class. accuracy (%)	96.05	96.53	97.85
F1-score(%)	94.64	95.16	95.97

(A)

	P	mono-B	bi-B
Type I err (%)	1.63	1.01	1.01
Type II err (%)	1.80	1.08	1.01
Class. err (%)	2.57	2.14	2.10
Type I err corrected (%)	67.47	90.01	88.67
Type II err corrected (%)	52.89	35.54	54.86
Class. accuracy (%)	96.05	97.06	97.32
F1-score(%)	94.71	95.81	95.89

(B)

	P	mono-B	bi-B
Type I err (%)	1.36	1.00	1.03
Type II err (%)	1.84	1.38	1.02
Class. err (%)	2.56	2.09	2.12
Type I err corrected (%)	66.54	73.61	72.69
Type II err corrected (%)	48.75	44.88	69.31
Class. accuracy (%)	96.08	96.89	97.27
F1-score(%)	94.85	95.74	95.83

(C)

TABLE IV
CLASSIFICATION ACCURACY OF MESSAGE EXTRACTION AT DECODER AVERAGED OVER 9 SEQUENCES AND 3 QUANTIZATION SCALES USING (A) RANDOM FORESTS (B) SVM-RBF (C) POLYNOMIAL CLASSIFIER

	P	mono-B	bi-B
Type I err (%)	1.58	1.01	1.02
Type II err (%)	1.84	1.51	1.01
Class. err (%)	2.56	2.16	2.08
Type I err corrected (%)	67.36	81.76	85.48
Type II err corrected (%)	52.00	42.82	72.46
Class. accuracy (%)	96.06	96.83	97.48
F1-score(%)	94.73	95.57	95.90

(A)

	P	mono-B	bi-B
Type I err (%)	1.99	1.13	2.54
Type II err (%)	5.76	1.43	21.07
Class. err (%)	2.63	2.14	3.26
Type I err corrected (%)	65.34	44.97	61.98
Type II err corrected (%)	59.39	59.16	37.55
Class. accuracy (%)	94.34	96.67	82.62
F1-score(%)	92.04	95.57	81.88

(B)

	P	mono-B	bi-B
Type I err (%)	2.28	1.28	3.91
Type II err (%)	1.00	1.00	1.01
Class. err (%)	3.56	2.21	2.09
Type I err corrected (%)	63.79	50.36	78.16
Type II err corrected (%)	100	97.69	90.12
Class. accuracy (%)	95.64	97.15	96.95
F1-score(%)	93.73	95.63	91.90

(C)

The second conclusion is that the quantization scale used in compressing the videos does not influence the proposed solution, meaning that the proposed solution is independent of the compression quantization scale, which is considered as an advantage. Therefore, for the experiments to follow, the average results using the three quantization scales are presented.

In Table IV, we repeat the experiment of Table III using two additional classifiers; SVM with an RBF kernel and a fifth order polynomial classifiers. The average results are shown for all video sequences and all quantization scales. For completeness, we show the average results of the Random Forest of Table III as well.

It is interesting to see that the average classification accuracy is 96.79% using Random Forests, 91.21% using SVM with FRB kernel and 96.58% using polynomial classifiers. The results show that SVM generated high classification results for P and mono-B-MBs which contain 4 classes only. On the other hand, it did not generate high classification results for bidirectional B-MBs that contain 16 classes. In fact, Type II error was around 21% on average, which is rather high.

In both P and mono-B MBs, the number of candidate MBs are 4, therefore, the classification results for P and mono-B are similar. On the other hand, in bi-B MBs, the number of candidate MBs are 16. Taking into account that only one candidate MB is labeled as the true MB (positive case) then for each MB classification there is 1 positive case and 15 negative cases resulting in a class imbalance. Random forests are known to work well in such cases and therefore all MB types generate similar results. This is not always the case for other classifiers, hence in Table IV, the SVM-RBF classifier generated a classification accuracy of around 83% for bi-B MBs.

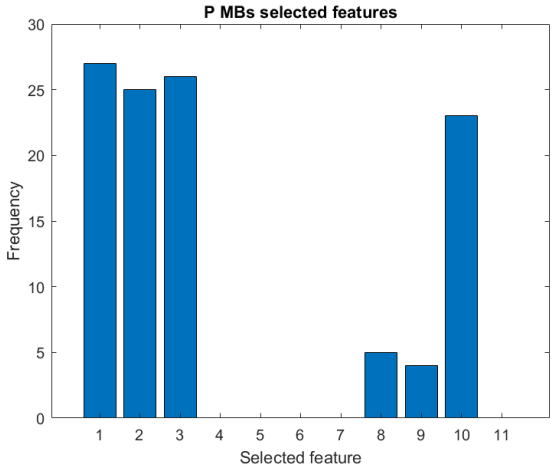
We also present the classification results using Random Forests with feature selection. The results in Table V are averaged over all test sequences and all quantization scales.

An observation from the results presented in Tables III ,IV and V is that the classification accuracies pertaining to P-MBs are slightly lower than their mono-B MBs counterparts. Although both MB types use 4 classification classes only, nonetheless, the mono-B MBs use reference frames (i.e. for motion compensation) that are one frame apart, whereas the P-MBs use reference frames that are 2 frames apart.

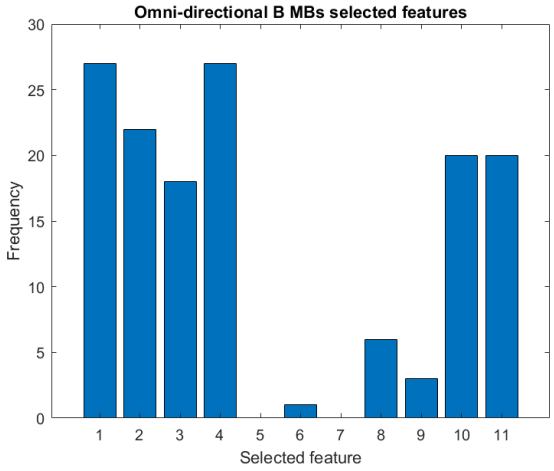
The classification results in Table V are very similar to the Random Forest results summarized in Table IV, part- A. Therefore, adding feature selection did not enhance the classification accuracy. Nonetheless, for completeness and proper analysis of the results, we examine the features that were selected per MB type.

In Fig. 5, we plot the histograms of the selected features per MB type.

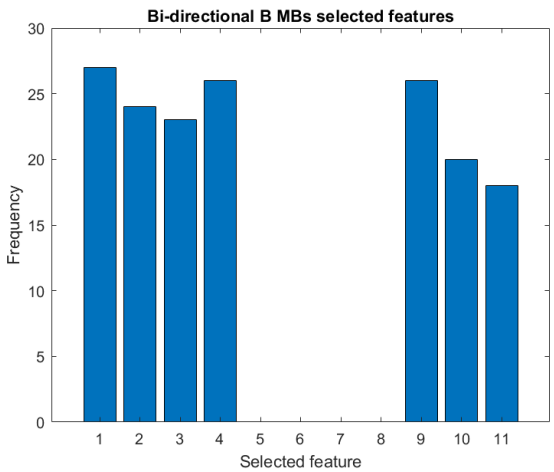
The description of the feature variables is in Table I. Note that feature variables with ID ‘4’ and ‘11’ do not apply to P-MBs as reported in part ‘a’ of the figure. Therefore, the corresponding bin value is zero. The histograms of part ‘a’ and part ‘b’ show that for P MBs and mono-directional B MBs, the most frequently used feature variables are the phase differences between the candidate MB and its top and left MBs of the same frame and the reference MB in the reference frame, and the sum of absolute values of the prediction errors. The same applies for part ‘c’ of the figure, which is related to bidirectional MBs, but with having feature 9 also as one of the most frequently selected features. According to Table I, this is the pixel variance of the reconstructed candidate MB.



(a)



(b)



(c)

Fig. 5. Histograms of the selected features per MB type.

We repeat the message embedding solution using a frame sequence of IPP…, without B-frames. Table VI presents the message classification results averaged over all video sequences.

The table shows that the overall average message classification accuracy using P frames only is 96.34% which is slightly lower than the case of using both P and B frames. This can be attributed to the fact that with IBP sequences, having bi-directional MBs results in improving the classification accuracy since the classifier in this case has more information to base its decision on, as the most influential features such as the MVs' phase differences and the sum of absolute values of the prediction errors are now
extracted in both directions; referencing to the previous and future frames.

TABLE V
CLASSIFICATION ACCURACY OF MESSAGE EXTRACTION AT DECODER AVERAGED FOR 9 SEQUENCES AND 3 QUANTIZATION SCALES USING RANDOM FORESTS WITH FEATURE SELECTION

	P	mono-B	bi-B
Type I err (%)	1.54	1.01	1.02
Type II err (%)	1.83	1.52	1.02
Class. err (%)	2.56	2.17	2.09
Type I err corrected (%)	68.66	77.55	82.18
Type II err corrected (%)	50.11	46.88	76.33
Class. accuracy (%)	96.06	96.82	97.48
F1-score(%)	94.72	95.64	95.91

The table also shows that at a low quantization scale with an IPPP… sequence, the average classification accuracy is nearly perfect. This is attributed to the fact that at low quantization scales, the image quality of reference frames used for image prediction and motion compensation is higher, therefore, the feature variables of motion compensated MBs are more representative.

TABLE VI
CLASSIFICATION ACCURACY OF MESSAGE EXTRACTION AT DECODER AVERAGED FOR 9 SEQUENCES WITH FRAME SEQUENCE OF IPP… USING RANDOM FORESTS

	Quant. scale 5	Quant. scale 15	Quant. scale 25
Type I err (%)	1.21	2.34	2.39
Type II err (%)	1.82	2.46	2.57
Class. err (%)	2.67	3.20	3.64
Type I err corrected (%)	79.21	89.63	88.38
Type II err corrected (%)	81.45	93.14	92.68
Class. accuracy (%)	96.74	96.39	95.89
F1-score(%)	94.47	94.46	91.45

In Table VII we list the processing time required to generate the Random Forests classification models per video sequence. We ran the experiments on a PC with Intel Core i7-37400QM, 2.7-GHz CPU with a 16-GB DDR3 RAM. As expected, the processing time increases with the increased spatial resolution of the videos. Note that data embedding and extraction in the context of this work need not be implemented in real-time.

TABLE VII
AVERAGE COMPUTATIONAL REQUIRED FOR RF MODEL GENERATION PER SEQUENCE.

Seq. ID	Resolution	Time(s)
1	416x240	1.85
2	416x240	2.03
3	416x240	1.8
4	832x480	6.82
5	832x480	4.67
6	832x480	3.0
7	1280x768	6.84
8	1280x768	9.92
9	1280x768	9.50

Moreover, as mentioned in the proposed solution of Section IV, Type I and II errors are dealt with by examining the boundary SADs between a candidate MB and its top/left previously decoded MBs.

It would be interesting to look at the classification results using this post processing technique only (as opposed to classification followed by post processing). Such an experiment would verify the importance of the proposed solution.

In Table VIII, we repeat the results reported in Table VI using post processing only for the identification of the original non-rotated MVs. The rather low classification accuracies justify the use of the proposed solution which is classification followed by post processing. The results in Table VIII show that the average classification accuracy is around 75% which is considered very low in this application. It is also shown that by increasing the quantization scale, the use of the SAD becomes less reliable as more DCT coefficients are quantized to zero. For example, the average classification accuracy dropped from 79% to 73% for the quantization scales of 5 and 25 respectively.

Going back to the discussion of the proposed solution, we examine the true average number of message bits per MB as presented in Table VIII.

Theoretically, a MB with one motion vector can embed 2 message bits and a MB with two MVs can embed 4 bits.

However, in practice there are 3 additional percentages that should be taken into account when computing the true average number of message bits per MB. These contain the percentage of intra-coded MBs, the percentage of MBs predicted with nil motion vectors and the percentage of boundary MBs that are not used for message embedding in this solution.

TABLE VIII
CLASSIFICATION ACCURACY OF MESSAGE EXTRACTION AT DECODER AVERAGED FOR 9 SEQUENCES WITH FRAME SEQUENCE OF IPP… USING POST PROCESSING ONLY

	Quant. scale 5	Quant. scale 15	Quant. scale 25
Avg. Accuracy	79.1%	75.1%	73.1%

TABLE IX
AVERAGE NUMBER OF MESSAGE BITS PER MB FOR EACH TEST SEQUENCE AVERAGED OVER ALL MB TYPES AND 3 QUANTIZATION SCALES

Seq. ID	IBP…	IPPP…
1	2.13	1.53
2	1.40	0.87
3	1.37	0.85
4	2.22	1.63
5	1.32	0.83
6	1.29	0.96
7	1.76	1.40
8	2.25	0.99
9	1.38	0.59
Avg.	**1.68**	**1.07**

Table IX shows that the true average number of message bits per MB varies from one video sequence to the other according to the above-mentioned three additional percentages. Therefore, video sequences with relatively high percentage of intra-coded MBs and/or high percentage of nil motion vectors, have fewer number of message bits per MB. Overall, the averages range from 1.29 to 2.25 bits per MB for an IBPBP… sequence and range from 0.59 to 1.63 bits per MB for an IPPP… sequence. In the latter case, since the reference frames used for prediction are one frame apart, it follows that the percentage of nil MVs are higher. This also contributed to the lower number of message bits per MB in videos with frame IPPP… sequences.

To examine the effect of the message classification accuracy on the quality of the reconstructed video, we report the PSNR differences between the video decoded normally without scrambling and without data embedding and the decoded video with scrambling and the proposed data embedding solution. The actual PSNR values of the decoded videos using the proposed solution are reported as well. The results are shown in Table X without the use of intermedia I-frames.

The loss in PSNR is due to the fact that the classification of the motion vectors is above 95% accurate, however it is not perfect. This imperfection results in few wrongly rotated MVs that implies inaccurate reconstruction of the underlying MBs. The inaccuracy of motion compensation results in higher error propagation if intermediate I frames are not present in the video bitstream, which is the case in our experiments. Therefore, the average drop in PSNR in such a case is 6.43 dB.

In summary, the Random Forest classifier generated the best classification results for the three types of MBs and the three levels of the quantization scale. The classification accuracy ranged from 96% to 97.5% and the average number of embedded bits are per MB are 1.68 and 1.07 with and without the use of B-frames respectively. However, these results are achieved at the expense of degrading the video quality at the decoder side as the classification accuracy is not 100%.

TABLE X
PSNR OF THE DECODED VIDEO WITHOUT USING THE PROPOSED ENCODER-DECODER SOLUTION. THE RESULTS ARE AVERAGED OVER 3 QPS AND 3 MB TYPES; P-MB, MONO-B-MB AND BI-B-MB

Seq. ID	PSNR Drop [dB]	Original PSNR [dB]
1	8.94	30.94
2	3.12	30.3
3	5.10	33.0
4	7.18	31.17
5	7.99	31.84
6	6.06	32.76
7	5.22	33.23
8	7.35	36.77
9	6.87	32.64
Avg.	**6.43**	**32.52**

To eliminate the effect on the PSNR of the reconstructed video, the proposed combined encoder-decoder solution was implemented. As mentioned in Section V, the training and classification processes are also performed at the encoder, so that the misclassified MBs are marked as non-embeddable by setting their coding mode to intra or no compensation, i.e. coded with zero MV. It is clear that interfering with the coding decisions at the encoder results in higher bitrate. Likewise, marking non-embeddable MBs results in lower message embedding rates. Table X reports the results for each test sequence averaged over all QPs for an IBPBP… sequence.

As shown in Table XI, in comparison with regular encoding, the average PSNR dropped by 0.12 dB and the average increase in bitrate is 6.2%. These numbers are at the encoder side.

At the decoder's side, the average extracted message bits dropped from 1.68 to 1.61 bits/MB as non-embeddable MBs are excluded from message embedding in this solution. However, this solution results in accurate decoding as the MVs of all MBs are correctly counter-rotated at the decoder with 100% classification accuracy and no picture drift.

TABLE XI
DROP IN PSNR, INCREASE IN BITRATE OF THE DECODED VIDEO AND THE EMBEDDING CAPACITY OF MPEG2 CODEC AS A RESULT OF THE COMBINED ENCODER-DECODER SOLUTION. THE RESULTS ARE AVERAGED OVER THE 3 QPS

Seq. ID	PSNR drop [dB] **(at encoder)**	Bitrate increase **(at encoder)**[%]	Bits /MB
1	0.09	6.63	2.06
2	0.06	2.72	1.32
3	0.26	8.15	1.28
4	0.14	12.87	2.15
5	0.12	5.93	1.24
6	0.11	5.90	1.23
7	0.05	1.88	1.69
8	0.17	8.38	2.18
9	0.08	2.96	1.31
Avg.	**0.12**	**6.16**	**1.61**

For comparison with existing work, namely, [7], [11], [19] and [20], we have identified 4 relevant papers that embed data in scrambled or encrypted video. Table XII below lists the reported average data embedding rate and drop in PSNR in dB.

TABLE XII
DATA EMBEDDING RATE AND DROP IN PSNR, COMPARISON WITH EXISTING SOLUTIONS

Solution	Bits/MB	PSNR drop [dB]
Proposed	1.61	0.00
Ref.[19]	0.75	0.51
Ref. [7]	1.06	0.20
Ref. [11]	0.05	0.20
Ref. [20]	0.08	0.17

As shown in the table, in comparison to [7], the proposed solution has a higher data embedding rate and a lower drop in PSNR. The work in [11] and [20], reported low PSNR drops of around 0.2 dB, however the embedding rates are in the range of 0.05-0.08 bit/MB which is considered very low in comparison to the proposed solution. Again, with the proposed encoder-decoder solution of Section V, the decoder has accurate reconstruction without any drop in PSNR. This is facilitated by marking non-embeddable MBs at the encoder as mentioned previously.

The computational complexity of the proposed solution is compared against the reviewed work as reported in Table XIII. The solutions reported in [7][11] and [20] have similar complexities as they reply on encryption and data embedding in codewords. The work in [19] on the other hand, requires manual intervention to locate regions of interest, it also requires a first round of fast decoding followed by full decoding. The proposed solution of this work embeds messages by rotating MVs and requires model generation using Random Forests. The time required to build the models is reported in Table VII above. Hence, the proposed solution has medium complexity when compared to other solutions, however, real-time data embedding and extraction is not a requirement in the proposed solution.

TABLE XIII
COMPLEXITY COMPARISON BETWEEN EXISTING WORK AND THE PROPOSED SOLUTION.

Solution	Comment on complexity
Reviewed [7], [11] and [20]	Selective encryption/decryption of video elements. Codeword substitution for data embedding. Complexity**:** Low
Reviewed [19]	Manual allocation of video areas to scramble. Two rounds of decoding required (with the first being fast) Optional encryption/ decryption of hidden data Complexity: Medium to high
Proposed	MV rotation for data embedding Machine learning needed to classify rotated MVs Complexity: Medium

Additionally, we compare our proposed solution against the work reported in [21], [22] and [23]. The authors in [21] reported the embedding capacity using a Hiding Ratio (HR) metric which is computed as:

$$HR = \text{length(message)} / \text{length(video)} * 100 \quad (7)$$

Where length(.) is a function that returns the number of bits. The work in [21] also reported the PSNR and the Bit Error Rate (BER) of the extracted embedded message. The work in [22] and [23] encrypted HEVC videos and embedded data in quantized DCT coefficients. The embedding rate was reported as an absolute number of hidden bits and in Kbit/s. In our work, the embedding rate measured in Kbit/s is computed as:

$$\text{Embedding\_rate} = (\text{avg\_msg\_bits\_per\_MB}) * (\text{MBs\_per\_frame}) * (\text{frames\_per\_sec})/1024 \quad (8)$$

The HR and the bitrate of the embedded messages using our proposed encoder-decoder solution are reported in Table XIV. The HR and message bitrate results vary per sequence according to the spatial solution of the underlying video frames. Larger frames have higher number of MVs and hence higher HR and higher message bitrates are expected. This is not always true as some video sequences might have a high percentage of skipped MBs, zero MVs or intra-coded MBs and therefore negatively affecting the HR and message bitrate as the case for Sequence 9 in the table.

TABLE XIV
MESSAGE HIDING RATIO AND BIT RATE OF THE ENCODER-DECODER SOLUTION AVERAGED OVER ALL QUANTIZATION SCALES

Seq. ID	f/s	HR(%)	Kbit/s
1	30	24.47	23.5
2	50	21.09	25.1
3	50	26.18	24.4
4	30	23.38	98.3
5	60	23.31	113.3
6	50	27.58	93.7
7	24	41.88	152.1
8	24	50.88	196.2
9	50	29.43	245.6
Avg.	**-**	**29.8**	**108**

In [21] the average reported HR was 1.46% and the Bit Error Rate (BER) of the embedded message was 0.023%. In contrast, the BER of the proposed solution is 0% for all sequences as the proposed encoder-decoder solution has a classification accuracy of 100% and therefore all embedded messages are extracted correctly. Additionally, the reviewed work reported PSNR results in the range of 70.4dB to Infinite. Our proposed solution on the other hand, does not result in any drop in PSNR at the expense of increased bitrate at the encoder side as elaborated upon previously.

The work in [22] encrypted HEVC syntax elements and embedded data in quantized transform coefficients. The average reported embedding rate was 3.3 Kbit/s. However, this was achieved at the expense of increasing the size of the bitrate by 7.8% and a drop of PSNR of 0.08 dB, with the use of periodic I-frames. Our proposed work reported an embedding rate of 108 Kbits without a drop in PSNR and without the use of periodic I frames. However, it caused a bitrate increase at the encoder's side by around 6%. The work in [23] extended and enhanced the work reported in [22]. The embedding rate was increased to 4.86 Kbit/s without increase in bitrate.

For completeness, we re-implemented the proposed solutions using the HEVC video coding standard. The same video sequences of Table II are encoded using an IPPP… sequence, with each sequence being compressed with four different quantization scales: 22, 27, 32 and 37. Tables XV and XVI illustrate the decoder-side solution results. Table XV shows the different error types and the classification accuracy averaged over all sequences using the Random Forests classifier. Table XVI shows the embedding rate for each test sequence averaged over all QPs. Using the HEVC codec, an average classification accuracy of 96.47% is reported with an average message embedding rate of 0.97 bits/MB. These results are very close to the results previously reported using the MPEG2 codec.

TABLE XV
CLASSIFICATION ACCURACY OF MESSAGE EXTRACTION AT HEVC DECODER AVERAGED FOR 9 SEQUENCES WITH FRAME SEQUENCE OF IPP… USING RANDOM FORESTS

	QP 22	QP 27	QP 32	QP 37
Type I err (%)	1.04	1.04	1.07	1.05
Type II err (%)	2.92	3.70	3.19	3.68
Class. err (%)	2.21	2.33	2.44	2.35
Type I err corrected (%)	83.72	89.43	89.70	83.28
Type II err corrected (%)	73.12	73.91	68.8	60.00
Class. accuracy (%)	96.83	96.59	96.45	96.00
F1-score(%)	94.74	94.10	94.19	94.27

TABLE XVI
AVERAGE NUMBER OF MESSAGE BITS PER MB FOR EACH TEST SEQUENCE AVERAGED OVER ALL 4 QUANTIZATION SCALES FOR HEVC

Seq. ID	IPPP...
1	1.43
2	0.73
3	0.74
4	1.57
5	0.77
6	0.43
7	0.88
8	1.45
9	0.68
Avg.	**0.96**

Table XVII repeats the results of Table XI using HEVC. The effect of the proposed combined encoder-decoder solution is examined using HEVC. Marking non-embeddable MBs at the encoder side resulted in an average PSNR drop of 0.14 dB and a 5.9% increase in the video bitrate.

TABLE XVII
DROP IN PSNR, INCREASE IN BITRATE OF THE DECODED VIDEO AND THE EMBEDDING CAPACITY OF HEVC CODEC AS A RESULT OF THE COMBINED ENCODER-DECODER SOLUTION. THE RESULTS ARE AVERAGED OVER THE 4 QPS

Seq. ID	PSNR drop (at encoder) [dB]	Bitrate increase (at encoder) [%]	Bits/MB
1	0.11	5.99	1.39
2	0.08	2.84	0.68
3	0.24	7.73	0.68
4	0.16	11.6	1.51
5	0.14	5.94	0.72
6	0.11	5.66	0.38
7	0.07	1.80	0.83
8	0.21	8.66	1.41
9	0.13	2.75	0.62
Avg.	**0.14**	**5.89**	**0.91**

Again, such an arrangement will allow the HEVC coder to correctly classify MVs, correctly decode MBs and extract message bits correctly without any picture drift at the expense of decreasing the average message embedding rate from 0.96 bits/MB, as reported in Table XV, to 0.91 bits/MB.

In summary, with the proposed encoder-decoder solution, the classification accuracy is increased to 100% which results in an unaltered quality of the decoded video. However, these results are achieved at the expense of increasing the bitrate at the encoder side, with around 6% increase for both MPEG2 and HEVC video bit streams. Additionally, the average number of embedded bits per MB are 1.61 and 0.91 with and without the use of B-frames respectively. In comparison to existing work as shown in Tables XII and XIV, the proposed work results in the best decoding quality and the highest message embedding rate.

VIII. CONCLUSION

A data embedding solution in scrambled video was proposed in this paper. The solution is based on rotating motion vectors according to message bits. A compliant decoder uses the first part of the video to generate a classification model to classify motion vectors into rotated or not. Once classified, the message bits are extracted, and the video is correctly reconstructed.

The proposed solutions were implemented using MPEG2 and HEVC video codecs. Detailed analysis of classification error was presented in terms of Type I and Type II errors. Such errors were detected and mitigated using a post-classification technique. It was also shown that the number of candidate MBs is irrelevant to the classification accuracy. The three MB types of P-MB, mono-B-MB and bi-B-MB had very similar classification accuracies. The average number of message bits embedded in rotated MVs reached up to 1.68 per MB.

The Random Forest classifier generated the best classification results in comparison to SVM and polynomial classifiers with a classification accuracy reaching to 97.8%.

As the classification accuracy was not 100%, the proposed solution has affected the PSNR of the reconstructed videos at the decoder. Therefore, combined encoder-decoder solution was proposed in which model generation and classification are also performed at the encoder to decide which MBs are eligible for data embedding based on whether they are classified correctly or not. This solution affected the compression efficiency as it interfered with mode decisions at the encoder. However, it resulted in 100% accurate classification and video reconstruction at the decoder without loss in PSNR.

In comparison to existing work, it was found that the proposed solution has the highest embedding capacity without any picture drift at the decoder side.

Future work includes studying the effect of the proposed solution on multilayer video coders. This is important as multilayer coders use inter layer prediction and possibly different MVs in different video layers. Future work also includes extending the proposed solution to sequence-independent learning, as the current feature extraction approach is specific to sequence-dependent learning. Lastly, the proposed system learning solution can be enhanced by periodically repeating the model generation every N frames or whenever a scene change is detected.

ACKNOWLEDGEMENT

The work in this paper was supported by a Faculty Research Grant from the American University of Sharjah (FRG19-S-E102). This paper represents the opinions of the authors and does not mean to represent the position or opinions of the American University of Sharjah.

REFERENCES

[1] T. Shanableh, “Data Hiding in MPEG Video Files Using Multivariate Regression and Flexible Macroblock Ordering," IEEE Transactions on Information Forensics and Security, 7(2), April, 2012.

[2] T. Shanableh, “Matrix encoding for data hiding using multilayer video coding and transcoding solutions,” Signal Processing: Image Communication, Elsevier, 27(9), October, 2012.

[3] T. Shanableh, “Altering Split Decisions of Coding Units for Message Embedding in HEVC,” Multimedia and applications, Springer, DOI: 10.1007/s11042-017-4787-6, May, 2017.

[4] K. Ma, W. Zhang, X. Zhao, N. Yu and F. Li, "Reversible data hiding in encrypted images by reserving room before encryption", IEEE Transactions on Information Forensics and Security, 8(3), March, 2013.

[5] S. Ong, K. Wong, K. Tanaka, “Scrambling–embedding for JPEG compressed image,” Signal Processing, vol. 109, 2015.

[6] W. Hong, T. Chen, H. Wu, "An improved reversible data hiding in encrypted images using side match", *IEEE Signal Process. Letters*, 19(4), April. 2012.

[7] D. Xu, R. Wang, Y. Shi, “Data hiding in encrypted H.264/AVC video streams by codeword substitution,” IEEE Transactions on Information Forensics and Security, 9 (4), 2014.

[8] Y. Yao, W. Zhang, N. Yu “Inter-frame distortion drift analysis for reversible data hiding in encrypted H.264/AVC video bitstreams,” Signal Processing, vol. 128, 2016.

[9] Y. Wang, M. O'Neill, F. Kurugollu, E. O'Sullivan, “Privacy region protection for H.264/AVC with enhanced scrambling effect and a low bitrate overhead,”, Signal Processing: Image Communication, vol. 35, 2015.

[10] Z. Ma, M. Jiang, H. Gao and Z. Wang, “Blockchain for digital rights management,” Future Generation Computer Systems, vol. 89, 2018.

[11] D. Xu, R. Wang, Y. Zhu, “Tunable data hiding in partially encrypted H.264/AVC videos,” Journal of Visual Communication and Image Representation, vol. 45, 2017.

[12] G. Correa, P. Assuncao, L. Agostini and L. da Silva Cruz “Fast HEVC Encoding Decisions Using Data Mining,” IEEE Transactions on Circuits and Systems for Video Technology, 25(4), April, 2015.

[13] E. Peixoto, T. Shanableh and E. Izquierdo “H.264/AVC to HEVC Video Transcoder based on Dynamic Thresholding and Content Modeling,” IEEE Transactions on Circuits and Systems for Video Technology, 24(1), January, 2014.

[14] M. Hassan and T. Shanableh, “Predicting split decisions of coding units in HEVC video compression using machine learning techniques,” Multimedia Tools and Applications, Springer, https://doi.org/10.1007/s11042-018-6882-8, November, 2018.

[15] L. Breiman, “Random Forests,” Machine Learning, 45(1), pp. 5–32, 2001.

[16] K. Toh, Q. Tran and D. Srinivasan, “Benchmarking a reduced multivariate polynomial pattern classifier,” IEEE Transactions on Pattern Analysis and Machine Intelligence, 26(6), pp. 740-755, 2004.

[17] F. Livingston, “Implementation of Breiman’s random forest machine learning algorithm,” ECE591Q Machine Learning Journal Paper, 2005.

[18] C.-C. Chang C.-J. Lin "LIBSVM: A library for support vector machines" ACM Trans. Intell. Syst. Technol. 2(3), pp. 27-53, April, 2011.

[19] P. Su, T. Tsai and Y. Chien, “Partial frame content scrambling in H.264/AVC by information hiding,” Multimedia Tools and Applications, 76(5), pp. 7473–7496, March, 2017.

[20] D. Xu, and R. Wang, “Context adaptive binary arithmetic coding-based data hiding in partially encrypted H.264/AVC videos,” Journal of Electronic Imaging 24(3), May, 2015.

[21] R. Patel K. Lad, M. Patel, et al., “A hybrid DST-SBPNRM approach for compressed video steganography,” Multimedia Systems (2021). https://doi.org/10.1007/s00530-020-00735-9

[22] D. Xu, “Commutative encryption and data hiding in HEVC video compression," IEEE Access, vol. 7, 2019.

[23] B. Guan, D. Xu and Q. Li, “An Efficient Commutative Encryption and Data hiding Scheme for HEVC Video,” IEEE Access, April, 2020

Afaf Eltayeb received her B.Sc. in electrical and electronics engineering, telecommunication discipline in 2016 from University of Khartoum, Sudan. She started her career by working as a teaching assistant at her undergraduate university, UofK. In 2017, she started working at Nile Centre for Technology and Research as a network and systems administrator after which she moved to Sudatel Company for Mobile Networks and worked as a Radio Network Optimization (RNO) engineer until December 2018.

From spring 2019 to fall 2020, she was granted an assistantship in the American University of Sharjah (AUS) for pursuing her M.Sc. in computer engineering beside working as a teaching and research assistant. Her research interests are computer and network security, video and image processing, IoT and machine learning.

Tamer Shanableh a senior member of the IEEE and a professional engineer. Was born in Scotland, UK in 1972. He studied at the University of Essex where he received his Ph.D. in electronic systems engineering in 2002 and his MSc in software engineering in 1998.

He then worked as a senior research officer at the University of Essex for three years, during which, he collaborated with BTexact on inventing video transcoders. He then joined

Motorola UK Research Labs and 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 a professor of computer science. During the summer breaks, Dr. Shanableh worked as a visiting professor at Motorola Labs in five different years. He spent the spring semester of 2012 as a visiting academic at the Multimedia and Computer Vision and Lab at Queen Mary, University of London, U.K. Dr. Shanableh authored more than 80 publications and has 6 patents. His research interests include digital video processing and pattern recognition.