Computers, Materials & Continua CMC, vol.63, no.3, pp.1205-1219, 2020

A MV-Based Steganographic Algorithm for H.264/AVC without Distortion

Hongqiong Tang[1, 2], Xiaoyuan Yang[1, 2, *], Yingnan Zhang[1] and Ke Niu[1, 2]

Abstract: H.264/AVC video is one of the most popular multimedia and has been widely used as the carriers of video steganography. In this paper, a novel motion vector (MV) based steganographic algorithm is proposed for the H.264/AVC compressed video without distortion. Four modules are introduced to eliminate the distortion caused by the modifications of motion vectors and guarantee the security of the algorithm. In the embedding block, the motion vector space encoding is used to embed a (*2n*+1)-*ary* notational number into an *n*-dimension vector composed of motion vectors generated from the selection block. Scrambling is adopted to disturb the order of steganographic carriers to improve the randomness of the carrier before the operation of embedding. The re-motion compensation (re-MC) block will re-construct the macroblock (MB) whose motion vectors have been modified by embedding block. System block plays the role of the generator for chaotic sequences and encryptor for secret data. Experimental results demonstrate that our proposed algorithm can achieve high embedding capacity without stego video visual quality distortion, it also presents good undetectability for existing MV-based steganalysis feature. Performance comparisons with other existing algorithms are provided to demonstrate the superiority of the proposed algorithm.

Keywords: Video steganography, motion vector, without distortion, H.264/AVC.

1 Introduction

Data hiding utilizes the redundancy of human sense organs on digital signals to embed secret data into digital host media without arousing suspicion. The application of data hiding in the video domain can coarsely be categorized as watermarking, steganography, error recovery (resilient), and general data embedding [Tew and Wong (2013)]. Video steganography has attracted extensive attention due to the development of video coding standards and network transmission. Video steganography includes three categories intra-embedding, pre-embedding and post-embedding [Liu, Liu, Wang et al. (2019)]. Intra-embedding methods combining with certain aspects of the compression characteristics utilize the coding process to hide information, such as intra-prediction [Nie, Xu, Feng et al. (2018); Li, Meng, Xu et al. (2019)], MVs [Niu, Yang and Zhang (2017)], quantized discrete cosine transformation (QDCT) coefficients [Ma, Li, Tu et al. (2010); Chen,

[1] Engineering University of PAP, Xi'an, 710086, China.

[2] Key Laboratory of Network and Information Security under the PAP, Xi'an, 710086, China.

[*] Corresponding Author: Xiaoyuan Yang, Email: yxyangyxyangthq@163.com.

Received: 10 February 2020; Accepted: 23 February 2020.

Wang, Wu et al. (2017); Zhang, Zhang, Yang et al. (2017)]. Pre-embedding methods Mstafa et al. [Mstafa and Elleithy (2016); Sadek, Khalifa and Mostafa (2017)] view the video as a sequence of motion pictures and embed the secret data into frame pixels directly with a certain intensity. While the post-embedding methods Xu et al. [Xu and Wang (2011)] mainly focus on the compressed bitstream, the computational complexity is low. The modern video coding standards H.26x and MPEG-x both have a high compression ratio, the redundancy has been eliminated to a great extent and the stream structure is more complex. As a consequence, it is difficult to embed data into the compressed video stream while maintaining good visual quality.

Motion vector usually has a wide range of value and is abundant in compressed video, it is considered to be ideal covert communication carriers by modifying the attributes or adjusting the motion estimation (ME) process. Early MV-based steganography selects a subset of MVs to be modified based on predefined rules. Jordan et al. [Jordan, Kuteter and Ebrahimi (1997)] first proposed using MVs to hide information by slightly compensating for parity. However, this algorithm ignores the influence of MV changes in the embedding process, resulting in a significant increase in the output bitrate of the payload video. Fang et al. [Fang and Chang (2006)] embedded the data into MVs by modulating the phase of MV. Xu et al. [Xu, Ping and Zhang (2006)] believed that the modification of MVs with small magnitude would result in deterioration in video compression efficiency. Hence the MVs with large magnitude are selected for data embedding. Aly [Aly (2011)] suggested tampering the MVs associated with high block prediction error by hiding one bit in each of their horizontal and vertical components. These algorithms have limited embedding efficiency and embedding capacity. To enhance the security and embedding efficiency of the algorithm, steganographic codes have been exploited to MV-based steganography. Hao et al. [Hao, Zhao and Zhong (2011)] proposed their steganography based on matrix encoding. Cao et al. [Cao, Zhao, Feng et al. (2011)] perturbed motion estimation was introduced by the wet paper code (WPC), and suboptimal MVs were selected for data embedding. Zhang et al. [Zhang, Cao and Zhao (2015)] proposed MV-based steganography with preserved local optimality. Yao et al. [Yao, Zhang, Yu et al. (2015)] defined a reasonable MV-distortion function by joining the spatial distortion change (SDC) and the prediction error change (PEC) together and then embedded data by two-layered Syndrome Trellis Code (STC). Yang et al. [Yang and Li (2018)] define motion vector space encoding combined with the original exploiting modification directions (EMD) for embedding data into H.265/HEVC videos. Zhai et al. [Zhai, Wang and Ren (2019)] first propose to embed in multi-domains for video steganography by combining partition modes (PMs) and MVs by multi-domain embedding (MDE) strategies.

In this paper, we proposed an MV-based steganographic algorithm for the H.264/AVC, which includes four modules: system block, selection block, embed block and re-MC block. System block takes the private key to generate three chaotic sequences. The first is used for encrypting and decrypting the secret data. The second is provided to the selection block, whose main function is to select qualified steganographic carriers under the control of chaotic sequences. The last one is used for scrambling to improve the randomness of the carrier modification. We take the motion vector space coding to slightly and efficiently modify an n-dimensional vector to embed a (2n+1)-*ary* notational

number. We introduced a re-MC block to re-reconstruct the MB whose MVs have been modified by the previous process. When compared with several related schemes, the proposed scheme not only holds excellent performance in visual distortion but also maintains a high embedding capacity.

The rest of this paper is organized as follows. Preliminaries and notations were described in Section 2, including motion estimation and motion vector space encoding. Detail descriptions of proposed algorithm are given in Section 3. Comprehensive experiments are conducted to show the performance of our algorithm in Section 4. Section 5 gives concluding remarks with some future research directions.

2 Preliminaries and notations

2.1 Data partitioning and motion estimation

The hybrid encoding process of the H.264 incorporates many new features including intra-prediction in intra-frame, multiple frame reference, quarter-pixel interpolation, and de-blocking filtering. The H.264 standard supports MVs of different block sizes and finer sub-pixels, i.e., 1/4 pixel resolution for luminance component. The MB of the I-frame only supports two partitions modes 16×16 and 4×4. Each MB of inter-frame can be divided into 16×16, 16×8, 8×16 and 8×8 according to encoding parameters, and the MB with partition mode 8×8 can further divide into four modes 8×8, 8×4, 4×8 and 4×4.

The motion estimation and compensation (MC) is a necessary procedure to reduce temporal redundancy. The schematic diagram of ME is shown in Fig. 1. The ME finds out the best-match block $\boldsymbol{R_B}$ for the current block $\boldsymbol{B}$ within a search area $\boldsymbol{R}_S$ in the reference frame $\boldsymbol{R}$. We can take the full search (FS) algorithm to search the global best-match block by exhaustively testing all the candidate blocks. Nevertheless, the FS algorithm holds huge computational complexity. There are many fast block-matching algorithms, Three-step search, four-step search, and diamond search to reduce computational complexity and find out the local best-match block. Mean squared error, sum of absolute differences (SAD) and mean absolute difference (MAD) are usually considered as matching criteria to calculate the prediction error of target block $\boldsymbol{B}$. Take SAD for example, $SAD(\boldsymbol{B}, \boldsymbol{R_B}) = \sum_{i=1}^{M} \sum_{j=1}^{N} |\boldsymbol{B}(i,j) - \boldsymbol{R_B}(i,j)|$, where $\boldsymbol{B}(i,j)$ and $\boldsymbol{R_B}(i,j)$ are luminance component pixel values of current block $\boldsymbol{B}$ and reference block $\boldsymbol{R_B}$, M and N are the width and height of $\boldsymbol{B}$ respectively. The main task of ME is to find a match block as local best-match block $\boldsymbol{R_B}$ to meet minimum prediction error, $\boldsymbol{R_B} = \underset{\boldsymbol{R_i} \in R_C}{\operatorname{argmin}} SAD(\boldsymbol{B}, \boldsymbol{R_i})$, where R_C is the set of all the candidate blocks within the search area $\boldsymbol{R_S}$ in the reference frame. After this step, the prediction error $\boldsymbol{P_E} = \{e(i,j) | i \in \{1..M\}, j \in \{1...N\}, e(i,j) = \boldsymbol{B}(i,j) - \boldsymbol{R_B}(i,j)\}$ will suffer integer transform and quantization to get QDCT coefficients. When finding out the best-match block $\boldsymbol{R_B}$ within the search area, we can get the MV $mv_B = (mv_x, mv_y)$, where mv_x and mv_y are horizontal and vertical components of mv_B respectively. For the decoder, the predicted value of the current block $\boldsymbol{P_B}$ can be acquired by mv_B from the reference frame. The final block is resumed by predicted value $\boldsymbol{P_B}$ and its corresponding prediction error $\boldsymbol{P_E}$ jointly.

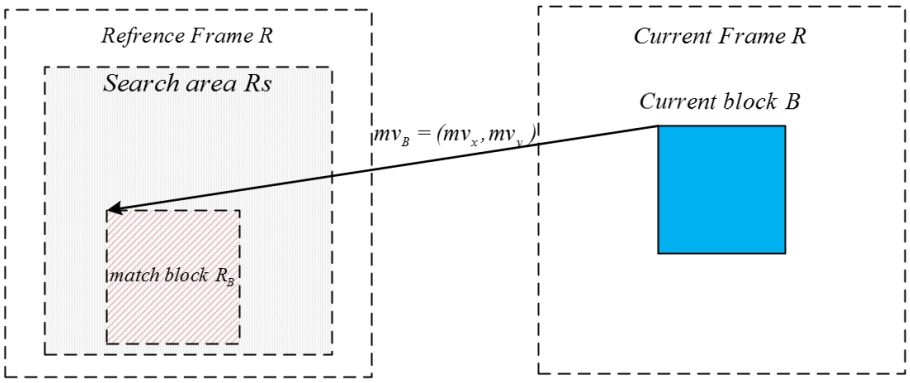

Figure 1: The schematic diagram of motion estimation

2.2 Motion vector space encoding

Assuming that an inter-MB in H.264 has n MVs, and each MV mv_i includes horizontal component mv_{xi} and vertical component mv_{yi}, $mv_i = (mv_{xi}, mv_{yi})$, where $i \in \{1,2,\ldots,n\}$. Then, we can obtain a vector with N elements expressed as: $\boldsymbol{C} = (c_1, c_2, \ldots, c_N)$, where $N \leq 2n$, $c_1 = mv_{x1}$, $c_2 = mv_{y1}, \ldots$, $c_{N-1} = mv_{xn}$, $c_N = mv_{yn}$. For each element of $\boldsymbol{C}$ c_i, we can map c_i to $x_i = f(c_i)$ from c_i and obtain another N-tuple $\boldsymbol{\tau} = (x_1, x_2, \ldots, x_N)$, where $f(\cdot)$ is defined as:

$$x_i = f(c_i) = (c_i mod(2N+1) + 2N + 1) mod(2N+1) \quad (1)$$

Now define an extract function F as a weighted sum modulo $(2N+1)$:

$$\mathrm{F}(x_1, x_2, \ldots, x_N) = [\textstyle\sum_{i=1}^{N}(x_i \cdot i)]\ mod(2N+1) \quad (2)$$

We can construct an N-dimensional space based on $\boldsymbol{\tau}$ and x_i is the corresponding ith dimensional coordinate of space. The space is denoted as $\boldsymbol{\Gamma}$ and called motion vector space. Any vector $\boldsymbol{\tau}$ can be mapped as a point in space $\boldsymbol{\Gamma}$ and the value is defined as $F(x_1, x_2, \ldots, x_N)$, which is an integer within $[0,2N]$.

For any point $\boldsymbol{P} = (x_1, x_2, \ldots, x_N)$ in $\boldsymbol{\Gamma}$, $\boldsymbol{P_i^+} = (x_1, x_2, \ldots, x_i^+, \ldots, x_N)$ and $\boldsymbol{P_i^-} = (x_1, x_2, \ldots, x_i^-, \ldots, x_N)$ are neighbor node of $\boldsymbol{P}$ in the ith dimension, where $x_i^+ = (x_i + 1) mod(2N+1)$ and $x_i^- = (x_i - 1) mod(2N+1)$. The neighbor node set with $2N$ elements denoted as $\boldsymbol{N_p} = \{\boldsymbol{P_1^+}, \boldsymbol{P_1^-}, \boldsymbol{P_2^+}, \boldsymbol{P_2^-}, \ldots, \boldsymbol{P_N^+}, \boldsymbol{P_N^-}\}$. There is a very important theorem that needs to be described in detail.

Theorem *For any point* $\boldsymbol{P} = (x_1, x_2, \ldots, x_N)$, *the mapping value of* $\boldsymbol{P}$ *and its all neighbor nodes are different from each other and the mapping value of these points can form a consecutive integer set of* $\{0,1,\ldots,2\mathrm{N}\}$.

Proof. Assume the mapping value of $\boldsymbol{P} = (x_1, x_2, \ldots, x_N)$ is a positive integer $H = F(\boldsymbol{P}) = F(x_1, x_2, \ldots, x_N) \leq 2N$. For any neighbor node P_i^+ and P_i^- we can get $H_i^+ = F(\boldsymbol{P_i^+}) = (H+i) mod(2N+1)$ and $H_i^- = F(\boldsymbol{P_i^+}) = (H-i) mod(2N+1)$. Finally, we can get a set $\boldsymbol{V}$ as follow:

$$\mathbf{V} = \{ (\mathrm{H} - \mathrm{N}) mod (2\mathrm{N} + 1), \dots , (\mathrm{H} - 1)\ mod (2\mathrm{N} + 1),\ (\mathrm{H}) mod (2\mathrm{N} + 1), (\mathrm{H} + 1) mod (2\mathrm{N} + 1), \dots, (\mathrm{H} + \mathrm{N}) mod (2\mathrm{N} + 1)\} \quad (3)$$

The set $\boldsymbol{V}$ consists of $2N + 1$ elements, and these elements are one-to-one corresponding to the elements in integer set $\{0,1,\dots,2N\}$ after the values of H and N determined. $(H) mod (2N + 1)$ is the mapping value of $\boldsymbol{P}$, $(H - N) mod (2N + 1)$ and $(H + N) mod (2N + 1)$ are the mapping value of $\boldsymbol{P_N^+}$ and $\boldsymbol{P_N^-}$ respectively. $(H - 1) mod (2N + 1)$ and $(H + 1) mod (2N + 1)$ are the map value of $\boldsymbol{P_1^+}$ and $\boldsymbol{P_1^-}$ respectively. So the theorem is proved to be correct.

Assume that there is a $(2N + 1)$ *-ary* notational number d and a point $\boldsymbol{P}$ with N coordinates. We want to modify at most one coordinate of $\boldsymbol{P}$ to $\boldsymbol{P'}$, which satisfies $d = F(\boldsymbol{P'})$. No modification is needed if d equals $F(\boldsymbol{P})$, $\boldsymbol{P'} = \boldsymbol{P}$. When $d \neq F(\boldsymbol{P})$, calculate $s = (d - F(\boldsymbol{P})) mod (2N + 1)$. If s is no more than N, increase the sth coordinate value by 1, that alters x_s to x_s^+, $\boldsymbol{P'} = (x_1, \dots, x_s^+, \dots, x_N)$, otherwise, alter x_{2N+1-s} to x_{2N+1-s}^-, $\boldsymbol{P'} = (x_1, \dots, x_{2N+1-s}^-, \dots, x_N)$.

3 Proposed scheme

In this section, we first generate three pseudo-random sequences using a chaotic system under the private key of the sender. The first pseudo-random sequence is used for the encryption of to-be-embedded data to ensure the security of secret data. The second one is used to determine whether the candidate MB is chosen as a steganographic carrier. The last and shuffle rule are combined to disturb the order of the steganographic carrier. Data embedding and extraction are given in subsections 3.2 and 3.3, respectively. The genetic structure of our proposed scheme for data hiding is shown in Fig. 2. Fig. 3 is the block diagram of the data extraction procedure.

3.1 Pretreatment

As shown in Fig. 2, the proposed algorithm includes four modules, the system block, the steganographic carrier selection block, the data embedding block, and the re-MC block. In the system block, a chaotic system is used to generate three pseudo-random sequences S_1, S_2 and S_3 under the private key K. For the chaotic system, we take the Logistic map, which is the most widely used nonlinear dynamic discrete chaotic map system and can be expressed as:

$$x_{n+1} = \lambda \cdot x_n \cdot (1 - x_n), 0 < \lambda \leq 4, x_n \in [0,1], n = 1,2, \dots \quad (4)$$

Any initial value of $x_0 \in (0,1]$ can generate a chaotic sequence $x_1, x_2, \dots, x_n, \dots$. while $\lambda \in (3.5699,4]$. Chaotic systems exhibit different characteristics with different λ and finally reach a chaotic state. The chaotic sequence needs to be transformed into a binary sequence by the following formula.

$$S(i) = \begin{cases} 0, & x_i \leq \alpha \cdot \overline{x} \\ 1, & x_i > \alpha \cdot \overline{x} \end{cases} \quad (5)$$

where $\overline{x} = \frac{1}{m}\sum_{i=1}^{m} x_i$, m is the length of the chaotic sequence generated by Eq. (4), $\alpha \in [0.5,1]$ is a weight. It should be noted that the parameters mentioned above include λ, x_0, α and m are generated under the control of the secret key K. Thus, S_1, S_2 and S_3 are finally

generated by Eqs. (4)-(5).

The stream encryption technology was used to encrypt the to-be-embedded data M to improve the security of data. The secret key of stream encryption S_1 comes from the binary sequences generated by the chaotic system. We can simply get the encrypted binary sequence M' through XOR (exclusive OR) operation $M'(i) = M(i) \oplus S_1(i)$.

3.2 Data embedding

In our proposed algorithm, MVs were chosen as the steganographic carrier. When the current block is decoding in the decoder, the MV and the reference frame of the current block are combined for prediction. The predicted value $\boldsymbol{P'_B}$ will not equal normal value $\boldsymbol{P_B}$ if the MVs have been revised, and the block $\boldsymbol{P'_B} + \boldsymbol{P_E}$ will produce dramatic changes in visual distortion, which is unacceptable for the steganographic algorithm. Therefore, the re-MC was introduced to re-acquire the real predicted value of $\boldsymbol{P'_B}$ based on the modified MV at the encoder and obtain the corresponding residual $\boldsymbol{P'_E}$.

The main function of the selection block is to select the steganographic carrier under the control of the secret key S_2. The MBs with the mode P8×8 are chosen as the candidates. It is based on the following three considerations for a trade-off between bit-rate, visual quality, and anti-steganalysis.

1) B frames are highly compressed in a bidirectional prediction manner, and within a high proportion in the compressed video stream. The modification of the MVs will cause a significant bit-rate variation.

2) There is a certain relationship between the mode and the texture complexity of the MB. MB with mode 16×16 is smoother than a 4×4 MB to a certain extent. The modification of MB with mode 16×16 will cause visual distortion easier.

3) The more steganographic carriers you select, the easier it will be for the steganalysis algorithm to detect them, so only the MB with mode P8×8 is selected.

Algorithm 1 Scramble process

Input: vector $\boldsymbol{C}$, secrete key $\boldsymbol{K}$

Output: disturbed vector $\boldsymbol{C'}$

len($\boldsymbol{v}$): get the length of vector $\boldsymbol{v}$

swap(a, b) : swap the value of a and b

read(**k**, n) : read a *n*-ary number from bitstream **k**

1. $\boldsymbol{C'} = \boldsymbol{C}$, N=*len*($\boldsymbol{C}$)
2. **for** *c* **in** $\boldsymbol{C'}$ **DO:**
3. *i*=*read*($\boldsymbol{K}$, N)
4. *swap*(*c*, $\boldsymbol{C'_i}$)
5. **end for**
6. **return** $\boldsymbol{C'}$

Algorithm 2 Embedding process

Input: disturbed vector $\boldsymbol{C}'$, secret data d

Output: modified vector $\boldsymbol{C}''$

len($\boldsymbol{v}$) : get the length of vector $\boldsymbol{v}$

$\boldsymbol{C}'' = \boldsymbol{C}'$, N=*len* ($\boldsymbol{C}'$)
construct $\boldsymbol{\tau} = (x_1, x_2, \ldots, x_N)$ from $\boldsymbol{C}'$
calculate D = $F(\boldsymbol{C}')$ and $s = f(d - D)$
if $s = 0$ **then**
  return $\boldsymbol{C}''$
else if $s < N$ **then**
  $\boldsymbol{C}''_s = \boldsymbol{C}''_s + 1$
else if $s \geq N$ **then**
  $\boldsymbol{C}''_{2N+1-s} = \boldsymbol{C}''_{2N+1-s} - 1$
return $\boldsymbol{C}''$

To make the selection of steganographic vectors random to prevent attacks from attackers. We introduce a pseudo-random sequence S_2 to determine whether the candidate MB carries secret data or not. If the current bit of S_2 is one, then the MB with mode P8×8 was chosen as the steganographic carrier.

First, the MVs of the MB selected as the steganographic carrier are grouped into an MV vector $\boldsymbol{C}$. We can get the scrambled vector $\boldsymbol{C}'$ using the Algorithm 1 under the pseudo-random sequence S_3. Second, we modify $\boldsymbol{C}'$ to $\boldsymbol{C}''$ to meet F($\boldsymbol{C}''$) = d, where d is a $(2N+1)$-*ary* notational number from encrypted binary sequence M', as shown in Algorithm 2. Finally, we restore the original order of the modified MV vector $\boldsymbol{C}''$, the entire embedding process finished.

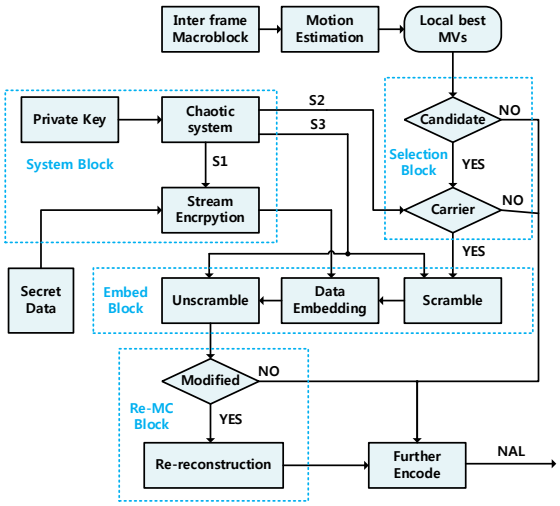

Figure 2: The genetic structure of our proposed scheme for data hiding

For example, assume $\boldsymbol{C} = \{1,2,-1,3\}, S_1 = 00101011, M = 10100110, S_3 = 1110110$.

So the encrypted binary sequence is $M' = M \oplus S_1 = 10001101$. We should disturb $\boldsymbol{C}$ before embedding data into it, then got disturbed carrier $\boldsymbol{C}' = \{2, -1,1,3\}$ according to Algorithm 1. We can construct $\boldsymbol{\tau} = (2,8,1,3)$ from $\boldsymbol{C}'$ and read a 9-*ary* notational number $d = 8$ from M' due to $N = 4$. The map value of $\boldsymbol{\tau}$ is $F(\boldsymbol{\tau}) = (1 \times 2 + 2 \times 8 + 3 \times 1 + 3 \times 3) mod 9 = 6$ is not equal to d, because $s = (d - F(\boldsymbol{\tau})) mod 9 = 2$ and $s < 4$, so $\boldsymbol{C}'' = \{2, -1 + 1,1,3\} = \{2,0,1,3\}$. Finally, we unscramble the MV vector $\boldsymbol{C}''$ and got $\{1,2,0,3\}$ under the control of S_3. From this simple example, we can see that 4-bit information is hidden by simply changing -1 to 0 with high embedding efficiency. It should be noted the maximum embedding rate can be achieved with $N = 2$.

3.3 Data extraction

Data extraction is the inverse process of data embedding under S_1, S_2, S_3. The entire detailed process is presented in Fig. 3. Firstly, the MBs with mode P8×8 selected from the compressed video stream. According to the pseudo-random sequence S_2, it can be known that the current MB contains secret data or not. If contained, the MV vector $\boldsymbol{C}$ is scrambled by Algorithm 1 under S_3. The process of extracting the secret data is straightforward, only by constructing $\boldsymbol{\tau}$ and then obtaining its mapping value d. Finally, we can extract all mapping value and group into an encrypted stream M', the embedded data stream M can be decrypted from M' by $M(i) = M'(i) \oplus S_1(i)$ with S_1.

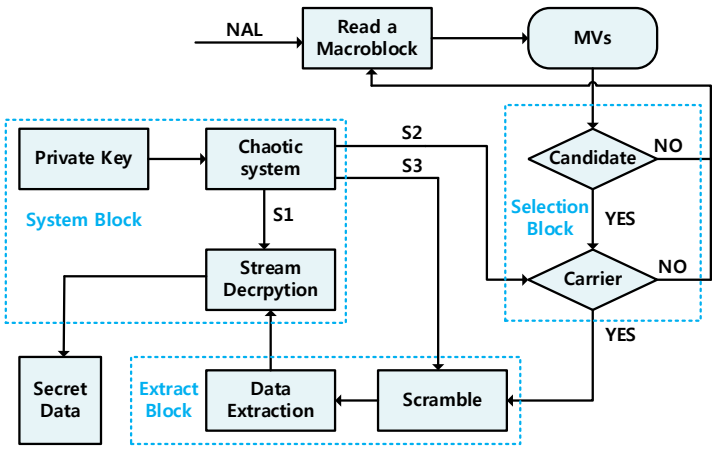

Figure 3: The block diagram of the data extraction procedure

For example, assume $\boldsymbol{C} = \{1,2,0,3\}, S_1 = 00101011, S_3 = 01110110$ and $\boldsymbol{C}$ is a carrier that is considered to contain secret information by Selection Block. So we disturb it and got $\boldsymbol{C}' = \{2,0,1,3\}$ under the secret key S_3. we can easily calculate the map value of $\boldsymbol{C}'$ equals to $d = F(\boldsymbol{C}') = (1 \times 2 + 3 \times 1 + 4 \times 3) mod 9 = 8 = b1000$, then use the secret key S_1 to decrypt d and we get the result is $b1000 \oplus b0010 = b1010$, which is same to M.

4 Experiment results and analysis

4.1 Experimental setting

Our experiment is implemented based on the H.264 reference encoder software JM 19.0. Six standard video sequences (QCIF, YUV4:2:0, 176×144) are used for experiments. The first 100 frames in test video are encoded at 30 frames/s with an intra-period equals nine. In the experiment, we inserted different numbers of B frames into two P frames to carry out experiments as almost all literature do not take B frames into account during the experiment, while the compressed video stream contains vast of B frames. The number of inserted B frames is represented as *Nbf*, which ranges from 1 to 5 in our experiment. Besides, we also did a comparative experiment under different quantization parameters (QP) values 24, 28, 32 and discussed it. The QP value for P-Slice, B-Slice and I-Slice is the same; the rest of the encoder parameters retained their default values. To comprehensively evaluate the performance of our proposed scheme, we set up different experiments to analyze visual quality, embedding capacity, bit-rate variation, and anti-steganalysis performance in the following subsections.

4.2 Visual quality

In this subsection, we introduce subjective and objective techniques to evaluate the visual quality of stego videos (H.264/AVC video with embedding data) and cover videos (standard H.264/AVC video without embedding data). Fig. 4 shows some decompressed frames on several stego video sequences under the condition *Nbf*=3, QP=28, all of them are P-frames. There are no significant visual distortions in these frames, subjectively, the proposed algorithm meets the requirement of imperceptibility.

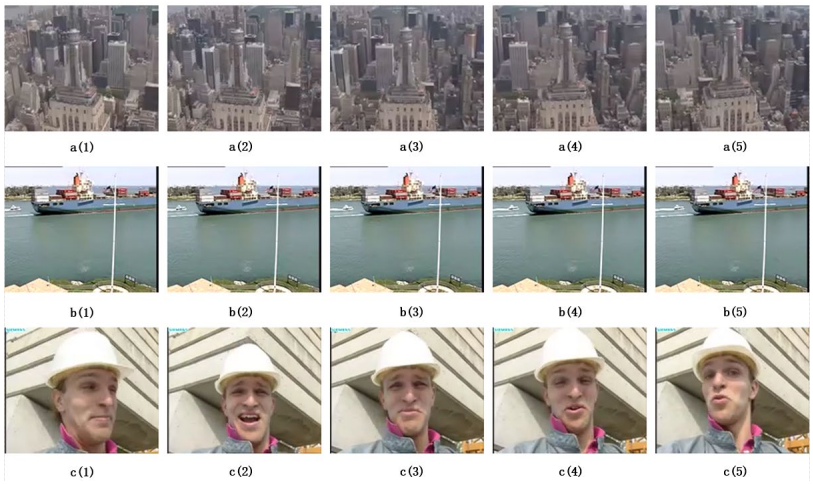

Figure 4: The visual quality of the 5th, 23th, 50th, 68th, 86th frames of stego video

Objective technologies are also taken to evaluate visual quality. Tab. 1 gives the average structural similarity (SSIM) and peak signal to noise ratio (PSNR) value of cover video and stego video. The value of PSNR and SSIM "original" is calculated between the original and cover videos; The "marked" is calculated between original and stego videos.

It is straightforward to see that the difference between original and marked is insignificant both PSNR and SSIM. The average PSNR degradation no more than 0.003 dB, it is worth noting that the average PSNR increased slightly when *Nbf*=5. Nevertheless, the average SSIM between marked and original are indistinguishable. The SSIM and PSNR for each frame of marked and original depicted by Fig. 5, which shows that the SSIM and PSNR for each frame of marked remain highly consistent with original, means without conspicuous differences. So the visual distortion performance of the proposed algorithm is superior.

Table 1: Average PSNR (dB) and SSIM

Seq	IBP				IBBBBBP			
	PSNR		SSIM		PSNR		SSIM	
	original	marked	original	marked	original	marked	original	marked
City	35.989	35.987	0.9596	0.9596	36.148	36.143	0.9609	0.9609
Foreman	36.418	36.418	0.9575	0.9572	36.534	36.531	0.9578	0.9579
Highway	37.943	37.939	0.9467	0.9467	37.975	37.979	0.9467	0.9468
News	37.077	37.070	0.9687	0.9687	37.446	37.459	0.9698	0.9699
Soccer	35.651	35.655	0.9249	0.9249	35.743	35.741	0.9262	0.9261
Mobile	33.777	33.767	0.9800	0.9800	33.869	33.869	0.9805	0.9805
Average	36.143	36.140	0.9562	0.9562	36.286	36.287	0.9570	0.9570

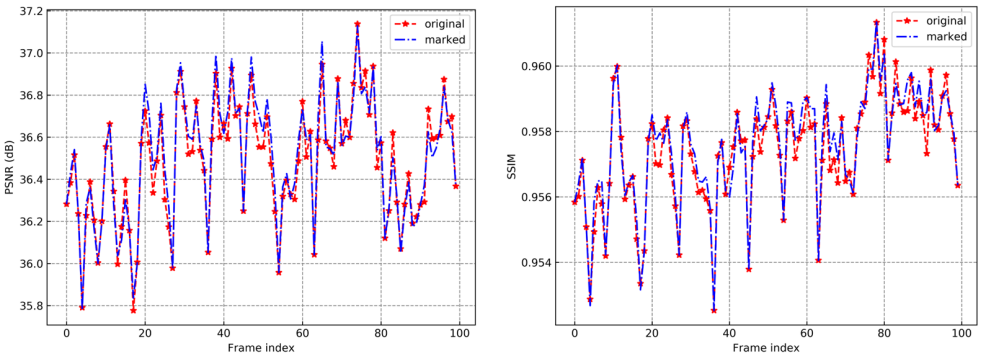

Figure 5: The variation of the SSIM and PSNR for foreman

4.3 Embedding capacity

The result of the PSNR difference (DPSNR) and the maximum embedding capacity shown in Tab. 2. The worst average embedding capacity can achieve 5087 bits while QP=32 and *Nbf*=3, but the average DPSNR is only 0.0070 dB. The best average embedding capacity can achieve 20327 bits while QP=24 and *Nbf*=1, the average PSNR difference is only -0.0021 dB that means the quality of the video was hardly affected by data embedding. By comprehensively analyzing the data, the embedding capacity decrease along with the increment of QP and *Nbf*. The reason for this phenomenon is the number of the macroblock with mode P8×8 declined as the lower compressed video

quality. Under the same QP conditions, different *Nbf* leads to different embedding capacity. The proposed algorithm shows good performance in terms of visual distortion and embedding capacity, and the embedding operation has almost no effect on the visual quality of stego videos.

Table 2: Average PSNR difference (dB) and the maximum embedding capacity (bits)

Seq		QP=24		QP=28		QP=32	
		DPSNR	capacity	DPSNR	capacity	DPSNR	capacity
City	IBP	-0.0230	21106	-0.0022	13197	0.0079	6228
	IBBBP	-0.0012	15417	-0.0069	9002	0.0154	4422
Foreman	IBP	0.0145	23068	0.0004	15322	0.0277	8663
	IBBBP	-0.0050	12346	0.0198	8111	0.0086	5248
Highway	IBP	-0.0076	10530	-0.0046	5363	-0.0005	2141
	IBBBP	0.0035	5937	0.0030	3564	0.0258	1506
News	IBP	-0.0105	12065	-0.0064	8383	-0.0205	5364
	IBBBP	0.0162	7735	-0.0028	5515	0.0088	3919
Soccer	IBP	0.0102	18692	0.0043	15308	0.0134	11701
	IBBBP	0.0037	6758	0.0018	5132	0.0066	4058
Mobile	IBP	0.0038	36504	-0.0098	30551	0.0139	21218
	IBBBP	0.0015	21413	0.0012	18286	-0.0004	11370
Average	IBP	-0.0021	20327	-0.0031	14687	0.0070	9219
	IBBBP	0.0031	11601	0.0027	8268	0.0108	5087

4.4 Bitrate variation

BRI is defined as following to evaluate the bitrate performance of our proposed algorithm further.

$$BRI = \frac{BR_{marked} - BR_{original}}{BR_{original}} \times 100\% \tag{6}$$

where BR_{marked} is the bitrate of the stego video, and $BR_{original}$ is the bitrate of the cover video. Tab. 3 shows that the average BRI value with different QP and *Nbf*, the maximum average BRI is no more than 0.538% and the minimum only 0.205%. As can be seen from Section 3.2, the data embedding process holds a high embedding efficiency, so the MBs that need to be reconstructed will also be reduced and the bit growth will be reasonable.

Table 3: The average BRI (%) with different QP and *Nbf*

QP	*Nbf*=1	*Nbf*=2	*Nbf*=3	*Nbf*=4	*Nbf*=5
QP=24	0.509	0.384	0.339	0.257	0.127
QP=28	0.551	0.377	0.547	0.219	0.260
QP=32	0.554	0.466	0.373	0.139	0.278
Average	0.538	0.409	0.420	0.205	0.222

4.5 Performance comparison

In this subsection, we mainly compare the proposed algorithm with the other three algorithms. Zhu et al. [Zhu, Wang and Xu (2010)] proposed that the to-be-embedded data was embedded by modulating the best search point during the motion of the quarter pixel. Su et al. [Su, Zhang, Zhang et al. (2014)] introduce the diamond coding to achieve information hiding by a slight modulation of MVs. To show the advantages of our algorithm, here is a literature that uses a similar method to take some modification only on partly middle- and high-frequency nonzero QDCT coefficients [Chen, Wang, Wu et al. (2017)] instead of MVs. Due to each steganographic algorithm has its unique properties, so it is difficult to provide an accurate comparison. Thus, we collected some related experimental data include the difference of PSNR (DPSNR), Capacity, and BRI for test video some video sequence. From Tab. 4, we hold a significant superiority in the DPSNR indicator compared with Zhu et al. [Zhu, Xu and Wang (2010)], Su et al. [Su, Zhang, Zhang et al. (2014)] and Chen et al. [Chen, Wang, Wu et al. (2017)]. The embedding capacity of the proposed scheme is less than Zhu except for video News, but generally better than Su and Chen. Since the focus of Chen's is the bitrate growth control of stego video, all other algorithms include ours are less than it, but the BRI of proposed is better than Zhu's and Su's. By contrast, we can confidently conclude that our scheme not only holds an excellent PSNR performance but also has a low BRI and high embedding capacity performance.

4.6 Anti-steganalysis performance

In this subsection, some experiments have been conducted to evaluate the anti-steganalysis performance of our steganographic algorithm. There are many steganalysis methods against MV-based steganography, such as AoSo (adding or subtracting one) [Wang, Zhao and Wang (2014)] and NPELO (near-perfect estimation for local optimality) [Zhang, Cao and Zhao (2016)]. The GOP (group of picture) level feature extraction is performed on each compressed video by AoSo and NPELO. For each kind of feature, we set up an experiment to evaluate anti-steganalysis performance. 60% of cover-stego pairs are selected randomly for training the SVMs (support vector machine) with Gaussian kernel using the LibSVM toolbox [Chang and Lin (2011)]. For simplicity, we select 1/5 of the feature of cover-stego pairs for classification. Five-fold cross-validation (CV) was taken to determine the optimal penalization parameter from set $\{10^{-3}, \ldots, 10^{5}\}$ before training each SVM. We choose the 10^{3} and 10^{5} as the penalty parameter for AoSo and NPELO feature classifier respectively. Each experiment repeated 10 times, and the average detection accuracy regarded as the final performance evaluation. The result is shown in Tab. 5. The average detection accuracy of AoSo and NPELO against our algorithm are 54.33% and 64.45% respectively. If we consider CSM (cover source mismatch), the real detection accuracy would be negatively influenced and maybe as bad as random guessing. Therefore, we have reason to believe that our embedding scheme has a good performance on anti-steganalysis.

Table 4: Performance comparison in terms of DPSNR (dB), embedding capacity (bits) and BRI (%) with QP=28

Seq		Chen's	Xu's	Su's	Proposed
Foreman	DPSNR	-4.71	-0.01	0.002	0.004
	Capacity	4059	18255	13000	15322
	BRI	0.27	1.31	4.52	0.90
Mobile	DPSNR	-2.61	-0.01	-0.014	-0.009
	Capacity	9669	32418	39357	30551
	BRI	0.12	0.46	4.46	0.70
Soccer	DPSNR	-1.63		-0.08	0.004
	Capacity	2300		18620	14308
	BRI	0.10		2.97	0.83
News	DPSNR	-2.64	0.00		-0.006
	Capacity	4309	7630		8383
	BRI	0.21	0.95		0.70

Table 5: The detection accuracy (%) of AoSo and NPELO against the proposed scheme

Feature	QP=24	QP=28	QP=32	Average
AoSo	55.92	53.75	53.33	54.33
NPELO	68.95	62.63	61.79	64.45

5 Conclusions

We have presented a novel MV-based steganographic algorithm to solve the visual distortion and capacity limitation for H.264/AVC. Four modules were introduced into the steganography to eliminate the distortion and ensure the security of the algorithm. The main idea of the paper is to use motion vector space coding to modulate the MVs to embed the secret information, and then re-reconstruct the MB to eliminate the distortion. As shown in the above experimental results, the steganographic algorithm can achieve good embedding capacity without visual distortion, the variation of bitrate for our algorithm also maintains a low-level increment, anti-steganalysis experiment results indicated that our algorithm has good performance against AoSo and NPELO feature.

In our further study, we will focus on the robustness of our algorithm, as we can see that our algorithm is fragile from the diagram, which has the same problems as almost all current steganographic algorithms, not means watermarking algorithm.

Funding Statement: This work was supported by the National Natural Science Foundation of China (Grant Nos. 61872384 and U1636114), and partly supported by the Natural Science Foundation of Engineering University of PAP (Grant No. WJY201915).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

Aly, H. A. (2011): Data hiding in motion vectors of compressed video based on their associated prediction error. *IEEE Transaction on Information Forensics and Security*, vol. 6, no. 1, pp. 14-18.

Cao, Y.; Zhao. X. F.; Feng, D. G.; Sheng, R. N. (2011): Video steganography with perturbed motion estimation. *Information Hiding 2011*, *Lecture Notes in Computer Science*, vol. 6958, pp. 193-207.

Chang, C. C.; Lin, C. J. (2011): LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*. vol. 2, pp. 1-27.

Chen, Y.; Wang, H. X; Wu, H. Z.; Liu, Y. (2017): An adaptive data hiding algorithm with low bitrate growth for H.264/AVC video stream. *Multimedia Tools and Applications*, vol. 77, no. 15, pp. 20157-20175.

Fang, D. Y.; Chang, L. W. (2006): Data hiding for digital video with phase of motion vector. *IEEE International Symposium on Circuit and Systems*, pp. 1422-1425.

Hao, B.; Zhao, L. Y.; Zhong, W. D. (2011): A novel steganography algorithm based on motion vector and matrix encoding. *3rd International Conference on Communication Software and Networks*, pp. 406-409.

Jordan, F.; Kutter, M.; Ebrahimi, T. (1997): Proposal of a watermarking technique for hiding/retrieving data in compressed and decompressed video. *Technical Report M2281*, *ISO/IEC Document*, *JTCI/SC29/WG11*.

Li, Z. H.; Meng, L. J.; Xu, Z. H.; Shi, Y. Q.; Liang, Y. C. (2019): A HEVC video steganalysis algorithm based on PU partition modes. *Computers*, *Materials and Continua*, vol. 59, no. 2, pp. 563-574.

Liu, Y. X.; Liu, S. Y.; Wang, Y. H.; Zhao, H. G.; Liu, S. (2019): Video steganography: a review. *Neurocomputing*, vol. 335, no, 8, pp. 238-250.

Ma, X. J.; Li, Z. T.; Tu, H.; Zhang, B. C. (2010): A data hiding algorithm for H.264/AVC video streams without intra-frame distortion drift. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 10, pp. 1320-1330.

Mstafa, R. J; Elleithy, K. M. (2016): Video steganography algorithm based on Kanade-Lucas-Tomasi tracking algorithm and error correcting codes. *Multimedia Tools and Applications*, vol. 75, no. 17, pp. 10311-10333.

Nie, Q. K.; Xu, X. B.; Feng, B. W.; Zhang, Y. (2018): Defining embedding distortion for intra prediction mode-based video steganography. *Computers*, *Materials and Continua*, vol. 55, no. 1, pp. 59-70.

Niu, K.; Yang, X. Y.; Zhang, Y. N. (2017): A novel video reversible data hiding algorithm using motion vector for H.264/AVC. *Tsinghua Science and Technology*, vol. 22, no. 5, pp. 489-498.

Sadek, M. M.; Khalifa, A. S.; Mostafa, M. G. M. (2017): Robust video steganography algorithm using adaptive skin-tone detection. *Multimedia Tools and Applications*, vol. 76, no. 2, pp. 3065-3085.

Su, Y. T.; Zhang, X. L.; Zhang, C. Q.; Zhang, J. (2014): Steganography algorithm based on motion vectors of H.264. *Journal of Tianjin University (Science and Technology) in Chinese*, vol. 47, pp. 67-73.

Tew, Y.; Wong, K. (2013): An overview of information hiding in H.264/AVC compressed video. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no, 2, pp. 305-319.

Wang, K. R.; Zhao, H.; Wang, H. X. (2014): Video steganalysis against motion vector-based steganography by adding or subtracting one motion vector value. *IEEE Transaction on Information Forensics and Security*, vol. 9, no. 5, pp. 741-751.

Xu, D. W; Wang, R. D. (2011): Watermarking in H.264/AVC compressed domain using Exp-Golomb code words mapping. *Optical Engineering*, vol. 50, no. 9, pp. 1-11.

Xu, C. Y.; Ping, X. J.; Zhang, T. (2006): Steganography in compressed video stream. *1st International Conference on Innovative Computing*, *Information and Control*, pp. 269-272.

Yao, Y. Z.; Zhang, W. M.; Yu, N. H.; Zhao, X. F. (2015): Defining embedding distortion for motion vector-based video steganography. *Multimed Tools and Applications*, vol. 74, no. 24, pp. 11163-11186.

Yang, J.; Li, S. B. (2018): An efficient information hiding method based on motion vector space encoding for HEVC. *Multimed Tools and Applications*, vol. 77, pp. 11979-12001.

Zhang, H.; Cao, Y.; Zhao, X. F. (2015): Motion vector-based video steganography with preserved local optimality. *Multimedia Tools and Applications*, vol. 75, no. 21, pp. 13503-13519.

Zhang, H.; Cao, Y.; Zhao, X. F. (2016): A steganalytic approach to detect motion vector modification using near-perfect estimation for local optimality. *IEEE Transaction on Information Forensics and Security*, vol. 12, no. 2, pp. 465-478.

Zhang, Y.; Zhang, M.; Yang, X.; Guo, D.; Liu, L. (2017): Novel video steganography algorithm based on secret sharing and error-correcting code for H.264/AVC. *Tsinghua Science and Technology*, vol. 22, no. 2, pp. 198-209.

Zhai, L. M.; Wang, L. N.; Ren, Y. Z. (2019): Multi-domain embedding strategies for video stegaography by combining partition modes and motion vectors. *IEEE International Conference on Multimedia and Expo*, pp. 1402-1407.

Zhu, H. L.; Wang, R. D.; Xu, D. W. (2010): Information hiding algorithm for H.264 based on the motion estimation of quarter-pixel. *2nd International Conference on Future Computer and Communication*, *IEEE*, pp. 423-427.