Circuits Syst Signal Process (2015) 34:3573–3594
DOI 10.1007/s00034-015-0024-6

An Information Hiding Approach Based on Integer Transform Coefficient and Virtual Space Encoding for H.264/AVC

Songbin Li[1] · **Jiangyun Fu**[1] · **Peng Liu**[1] · **Yuxin Jiang**[1]

Received: 24 June 2014 / Revised: 7 March 2015 / Accepted: 9 March 2015 /
Published online: 20 March 2015

Abstract This paper gives an information hiding method for H.264/AVC compressed video stream based on integer transform coefficient and virtual space encoding. In this method, the integer transform coefficients which are selected as carriers are divided into multiple groups. And it only needs to change at most the value of one of the coefficients in each group to embed $\mathrm{Log}_2(2N+1)$ binary bits when each group contains N coefficients. Moreover, the required change is simple, which can be achieved by adding 1 or -1 to the coefficient. Since only one coefficient's value is slightly changed for every $\mathrm{Log}_2(2N+1)$ secret bits, this method does less harm to the original encoded video stream caused by the information embedding compared with that by current methods. Furthermore, this method only chooses the macroblock that has rich texture as the candidate carrier macroblock, and it employs the embedding position template to select the candidate 4×4 coded blocks (I4-blocks) from the carrier macroblock for watermark bits embedding. In this way, the counteracting steganalysis performance is significantly improved. Experimental results on several test sequences demonstrate that the proposed approach can realize blind extraction and the performance is superior to that of the other current state-of-the-art information hiding methods also based on modulating integer transform coefficients.

✉ Songbin Li
lisongbin_work@126.com

Jiangyun Fu
cloud@fujiangyun.com

Peng Liu
liup@dsp.ac.cn

Yuxin Jiang
jiangyx@dsp.ac.cn

[1] Haikou Laboratory, Institute of Acoustics, Chinese Academy of Sciences, Haikou 570105, China

Keywords Information hiding · H.264/AVC · Integer transform coefficient · Embedding position template · Virtual space encoding

1 Introduction

Information hiding is the process of putting secret information into the carrier without perception. The carrier can be text, digital voices, digital image, video signal and so on. General model of information hiding system can be traced back to the prisoner's problem [19] and the subliminal channel [18,20] published in 1983 by Simmons.

As the Internet and multimedia technology develop, the transmission and exchange of text, audio, image, video and other multimedia information have become more and more convenient. Video information has characteristics of intuition, certainty, high fidelity, etc. Thus, the video data stream has gradually become an important component of the Internet network communication flow. This huge video flow can be easily taken as potential information hiding carrier, especially as the network streaming media service is evolving in recent years. Generally, video needs to be compressed and encoded before transmitting through the network. Therefore, the research of information hiding based on compressed video stream is gaining more and more attention in recent years. H.264/AVC is an important international video coding standard [26]. It is designed to cover the entire field of video applications and is being widely used now. So, how to hide information into the H.264/AVC encoded video stream has become a hot issue. Due to the lack of the redundancy of the H.264/AVC video stream, this issue is very challenging. Many scholars are dedicated into solving this problem.

A review of the current state-of-the-art information hiding methods for the H.264/AVC video stream is given in [22]. The existing information hiding methods for H.264/AVC standard mostly exploit the encoding characteristics of H.264/AVC, such as the smaller inter-prediction block, the unique intra-prediction mode, the integer DCT transform, the multiple reference frames and the adaptive entropy coding. According to the different encoding characteristics used for information hiding, these methods can be roughly divided into four major categories: (1) intra-frame prediction mode modulation information hiding methods [6,13,29,31]; (2) motion vector modulation information hiding methods [3,9,15,21]; (3) entropy encoding element modulation information hiding methods [11,14,24,27,30,33]; and (4) integer transform coefficient modulation information hiding methods [2,4,10,12,16,23,28,32]. There are also some other particular methods which has drawn little attention and cannot be divided into those four major categories, such as information hiding taking advantage of the different block sizes used by the H.264/AVC encoder [8], information hiding based on IPCM coding macroblock [1,7] and information hiding integrating several of the above methods [5,17].

The first category of methods embed secret bits based on modifying the intra-frame prediction mode. This type of method just varies in the modulating rule of prediction mode and the selection of I4-blocks which will be used for information embedding. The drawback of this type of method is their limited embedding capacity, because information hiding can only be done in the I-frames and the number of I-frames in a video sequence is very few. The second category of methods hide information by

changing the motion vectors of the encoded video stream directly or by modifying the search range of the optimal motion vector in the encoding process. Because the H.264/AVC video stream has a lot of motion vectors even at very low bit stream, this type of method possesses larger embedding capacity than the first type. Another advantage is that motion vectors exist in P and B frames and their deviation caused by the information hiding is unlikely to propagate. This will avoid significantly affecting the decoding video quality. However, this type of method is likely to introduce more bit rate increase, according to our previous study. The third category of methods embed information through modifying the CAVLC, CABAC or Exp-Golomb entropy coding elements. This kind of methods is very easy to implement. However, they are prone to degrade the decoding video quality seriously. As for the fourth category of methods, they hide information by modifying the H.264/AVC integer transform coefficients. For example, the coefficient is changed to an even or odd number, representing the embedding of a bit of "0" or "1", respectively.

Among all the state-of-the-art methods, the integer transform coefficient modulation (ITCM) information hiding approaches have some advantages. Integer transform will occur in all types of encoded video frames, so it has enough carriers for hiding secret bits. Besides, this type of method generally has low complexity and can be integrated into the encoding process. This property makes it especially suitable for embedding information during the real-time encoding process. Therefore, we are of the opinion that the ITCM embedding approaches are very promising development orientation for efficient information hiding in H.264/AVC video stream.

However, the carrier coefficient is probable to be changed in current ITCM information hiding methods. In other words, one coefficient will be modified (or not) when one watermark bit is hidden. Generally speaking, the ITCM information hiding will increase the bit rate of the compressed video stream and degrade the decoding video quality. Furthermore, this negative effect usually becomes more obvious as the number of the changed coefficients increases. Moreover, when the embedded secret information is greater than a certain threshold, the decoding video quality may deteriorate seriously and the coefficients' statistical characteristics may change. In this case, the information hiding operation will be at a high risk of being perceived by the attacker. Hence, how to improve current ITCM methods' embedding efficiency as well as the reduction in the total number of coefficients needed to be modified at the same embedding capacity becomes an important issue. The embedding efficiency means the number of hidden watermark bits per host coefficient (modified or not). In this paper, we aim to improve the current ITCM methods and give a more efficient embedding algorithm.

Although our algorithm implements the information hiding still by modifying integer transform coefficients similar to the current ITCM methods [2,4,10,12,16,23,28,32], our algorithm differs significantly from them. In fact, we do not modify the carrier coefficients directly; instead, we make the information hiding in a high dimensional virtual space. The virtual space is built through converting carrier coefficients set into coordinates of the virtual space according our mapping rules. With the introduction of the virtual space, for each coefficient group comprised of N carrier coefficients, we can embed $\mathrm{Log}_2(2N+1)$ binary bits into it. Moreover, our algorithm can guarantee that only one integer transform coefficient's value will be changed at most. For

current ITCM methods, only one secret bit can be hidden when one transform coefficient is changed. Because only one coefficient's value is slightly changed for every $\mathrm{Log}_2(2N+1)$ secret bits, our approach can guarantee better decoding video quality and smaller code rate increase caused by the information hiding than the current methods. Furthermore, our method only chooses the macroblock that has rich texture as the candidate carrier macroblock, and only some specific 4×4 coded blocks (I4-blocks) in the candidate macroblock can be selected for embedding watermark bit. Which of the I4-blocks will be selected is determined by the predefined embedding position template (EPT), and for different candidate macroblock, which EPT will be adopted is random. In this way, we effectively enhance the hiding method's counteracting steganalysis performance.

The rest of this paper is organized as follows. In Sect. 2, we briefly illuminate the integer transform of H.264/AVC encoder and the main technical details of information hiding methods based on modulating integer transform coefficients. And we present the information embedding and extracting technical detail of our proposed method in Sects. 3 and 4. Next, experimental results are presented in Sect. 5, and the conclusions are given in Sect. 6.

2 H.264/AVC Integer Transform and Transform-Based Information Hiding Methods

2.1 Integer Transform of H.264/AVC Encoder

H.264/AVC standard does not define the implementation of its encoder and decoder in detail. Instead, it only regulates the syntax of encoded video code stream. H.264/AVC encoder still adopts hybrid encoding method, and core process is transformation and prediction. The common structure (in other words, coding processes) of H.264/AVC encoder is shown in Fig. 1.

The input frame (or input field) F_n is processed in the unit of macroblock. In order to reduce data redundancy, each encoding block is first processed by the intra-frame or inter-frame prediction coding module. The intra-frame prediction is used for reducing

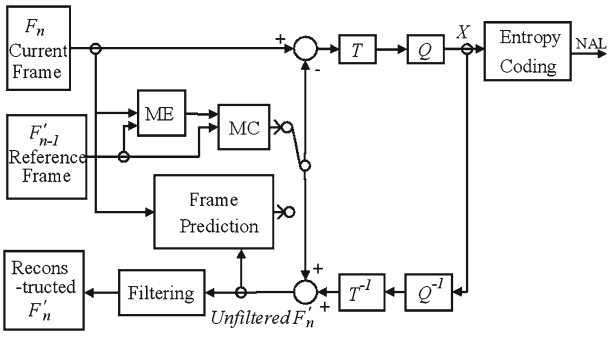

Fig. 1 Structure of H.264/AVC encoder

the data redundancy in the spatial domain, while the inter-frame prediction also called motion estimation (ME) and motion compensation (MC) is used for removing the data redundancy in the time domain. The prediction process will generate the prediction frame. When it is subtracted by the original frame, the encoder will generate the prediction residuals. With a view to further reduce the data redundancy, the residuals are transferred into a series of transform coefficients by the block transformation step T and quantization step Q. Finally, the series of transform coefficients will be converted into code elements by entropy encoding. Combining them with some side information (such as quantization parameter, prediction mode, motion vector), we can obtain the compressed video stream. And the stream can be transmitted and stored in the NAL (network adaptive layer) format. Besides, in order to generate the reference image for prediction, the encoder must be able to reconstruct the encoded frame through inverse quantization step Q^{-1} and block transformation step T^{-1}.

Discrete cosine transform (DCT) is commonly used in image coding, because it approximates to the optimal K–L transform in theory under certain conditions. It can effectively reduce the data redundancy. However, the original DCT operates on floating-point numbers. This will raise the complexity of the transform system and make the output coefficients of inverse transform different from the input. In order to avoid this issue, H.264/AVC introduces a novel transform: the integer transform. Actually, the integer transform is derived from DCT and called integer DCT transform. This novel transform realizes the approximate DCT only using integer addition, subtraction and shift operation.

The 4×4 integer DCT transform of H.264/AVC is defined by:

$$\begin{aligned} Y &= \left(C_f X C_f^T\right) \bigotimes E_f \\ &= \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} X \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix} \right) \bigotimes \begin{bmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{bmatrix} \end{aligned} \tag{1}$$

where X is the 4×4 residual matrix generated by intra- or inter-prediction; the symbol $\bigotimes$ denotes the coefficient product in the corresponding position of two matrixes; a and b are two constants. We can see that the core matrix C_f of the transform kernel only contains elements of ± 1 and ± 2, which can be realized only through adding, subtracting and shifting operations avoiding the time-consuming multiply operation in the original DCT. The element of E_f can be merged into the process of quantization. After quantization, we can obtain matrix Q as shown below:

$$Q = \left(C_f X C_f^T\right) \bigotimes \begin{bmatrix} \frac{a^2}{S} & \frac{ab}{2S} & \frac{a^2}{S} & \frac{ab}{2S} \\ \frac{ab}{2S} & \frac{b^2}{4S} & \frac{ab}{2S} & \frac{b^2}{4S} \\ \frac{a^2}{S} & \frac{ab}{2S} & \frac{a^2}{S} & \frac{ab}{2S} \\ \frac{ab}{2S} & \frac{b^2}{4S} & \frac{ab}{2S} & \frac{b^2}{4S} \end{bmatrix} - \begin{bmatrix} Q_0 & Q_1 & Q_2 & Q_3 \\ Q_4 & Q_5 & Q_6 & Q_7 \\ Q_8 & Q_9 & Q_{10} & Q_{11} \\ Q_{12} & Q_{13} & Q_{14} & Q_{15} \end{bmatrix} \tag{2}$$

where S denotes the quantization step of the coefficients, Q_0 in the matrix Q is the DC coefficient and others are the AC coefficients of the integer transform. The ITCM information hiding methods commonly embed secret bits into the matrix Q by modifying the coefficient Q_i according to some modulation rules.

2.2 Information Hiding Based on Modulating Transform Coefficients

Generally speaking, the ITCM information hiding methods all hide information by modifying the integer transform coefficient of H.264/AVC. The difference is only in the modulation rule. Among the current methods, the algorithm based on modulating coefficient's parity [10] is rather typical. This approach hides the secret information in the nonzero AC coefficients of matrix Q. Its embedding algorithm is defined as follows:

When $W_i = 0$

$$Q_i{}' = \begin{cases} Q_i & \text{if} \quad Q_i \text{Mod} 2 = 0 \\ Q_i + 1 & \text{if} \quad Q_i \text{Mod} 2 = 1 \\ Q_i - 1 & \text{if} \quad Q_i \text{Mod} 2 = -1 \end{cases} \tag{3}$$

When $W_i = 1$

$$Q_i{}' = \begin{cases} Q_i & \text{if} \quad Q_i \text{Mod} 2 = \pm 1 \\ Q_i - 1 & \text{if} \quad Q_i \text{Mod} 2 = 0 \& Q_i \geq 0 \\ Q_i + 1 & \text{if} \quad Q_i \text{Mod} 2 = 0 \& Q_i < 0 \end{cases} \tag{4}$$

where W_i is the watermark bit; Q_i is the selected coefficient from matrix Q defined in (2) before hiding secret bit, and it will turn into $Q_i{}'$ after watermark embedding. And this algorithm's hiding watermark bit extracting process is defined below:

$$W_i = \begin{cases} 0 & \text{if} \quad Q_i \text{Mod} 2 = 0 \\ 1 & \text{if} \quad Q_i \text{Mod} 2 = 1 \end{cases} \tag{5}$$

The method in [23] is also based on coefficient's parity, but its embedding rules are defined as follows:

When $W_i = 0$

$$Q_i{}' = \begin{cases} Q_i & \text{if} \quad Q_i \text{Mod} 2 = 1 \\ Q_i - 1 & \text{if} \quad Q_i \text{Mod} 2 = 0 \& Q_i \geq 0 \\ Q_i + 1 & \text{if} \quad Q_i \text{Mod} 2 = 0 \& Q_i < 0 \end{cases} \tag{6}$$

When $W_i = 1$

$$Q_i{}' = \begin{cases} Q_i & \text{if} \quad Q_i \text{Mod} 2 = 0 \\ Q_i + 1 & \text{if} \quad Q_i \text{Mod} 2 = 1 \& Q_i \geq 0 \\ Q_i - 1 & \text{if} \quad Q_i \text{Mod} 2 = 1 \& Q_i < 0 \end{cases} \tag{7}$$

Moreover, this algorithm only selects the last nonzero AC coefficient of matrix Q as the carrier coefficient. Another similar algorithm is proposed in [16]. This method hides secret bits as follows:

When $W_i = 0$

$$Q_i{}' = \begin{cases} Q_i - 1 & \text{if} \quad Q_i \text{Mod} 2 = 1 \\ Q_i & \text{if} \quad Q_i \text{Mod} 2 = 0 \end{cases} \tag{8}$$

When $W_i = 1$

$$Q_i{}' = \begin{cases} Q_i & \text{if} \quad Q_i \text{Mod} 2 = 1 \\ Q_i + 1 & \text{if} \quad Q_i \text{Mod} 2 = 0 \end{cases} \tag{9}$$

The method proposed in [4] first writes the address of the last nonzero coefficient Q_i of matrix Q into an array for each I4-block in a macroblock and then accumulates all the last nonzero coefficients of the macroblock to obtain information hiding flag s_i. The secret bit W_i is hidden according to the following formula:

$$Q_i{}' = \begin{cases} Q_i + 1 & \text{if} \quad s_i \text{Mod} 2 = 1 \& W_i = 0 \& Q_i \neq -1 \\ Q_i - 1 & \text{if} \quad s_i \text{Mod} 2 = 1 \& W_i = 0 \& Q_i = -1 \\ Q_i & \text{if} \quad s_i \text{Mod} 2 = 1 \& W_i = 1 \\ Q_i & \text{if} \quad s_i \text{Mod} 2 = 0 \& W_i = 0 \\ Q_i + 1 & \text{if} \quad s_i \text{Mod} 2 = 0 \& W_i = 1 \& Q_i \neq -1 \\ Q_i - 1 & \text{if} \quad s_i \text{Mod} 2 = 0 \& W_i = 1 \& Q_i = -1 \end{cases} \tag{10}$$

Other methods are essentially similar to the above methods with only nuance of implementation details. For example, the method in [12] sets a particularly strict limitation on the selection of the macroblocks along with the coefficients, and only the fitted macroblocks and coefficients can be taken for information embedding. The method in [28] hides the secret bit only in the DC coefficients of the intra-frame macroblocks, and the method in [2] determines whether to perform the embedding secret information into a given I4-block according to a given threshold. The method in [32] has some special feature. It transfers the common secret bit $\{0, 1\}$ into the bipolar watermark $\{-1, 1\}$ for hiding. And it only selects the host integer DCT coefficients from the texture-rich blocks and makes hiding by directly adding the bipolar watermark to the selected coefficients. In other words, the current ITCM methods can only embed one secret bit when one transform coefficient is changed, and we want to overcome this drawback and improve the ITCM method's embedding efficiency.

3 Information Embedding Process

Our method's information hiding operation is integrated into the H.264/AVC encoding process. The embedding process is shown in Fig. 2. The main component is the selection of carrier coefficients and the embedding of $2N + 1$-ary numbers. With a view to reduce the impact brought by embedding on the decoding video quality and boost the invisibility, we only select some I4-blocks from the macroblock that has rich texture to hide information. Further, for the host I4-blocks, we only choose some nonzero AC coefficients as embedding carriers. The secret bits are firstly encrypted using the user's private key. And then, the encrypted secret bits are converted to the $2N + 1$-ary

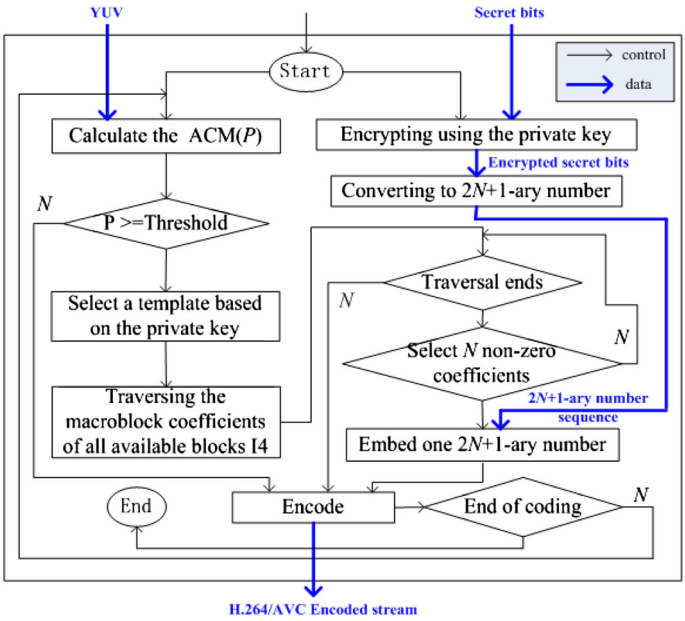

Fig. 2 Embedding process integrated into H.264/AVC encoder

number sequence. For example, if the secret bits is 1011 and $N = 2$, then we can get a quinary number sequence: 21; if $N = 3$, then we can get a septenary number sequence: 14. In our approach, the integer DCT coefficients selected as carriers are divided into groups where each group contains N coefficients. It only needs to change one integer DCT coefficient's value at most in each group to embed one $2N + 1$-ary number. Besides, it is a lightweight change where the coefficient only adds or subtracts 1. Next, we elaborate our method in detail.

3.1 The Selection of Carrier Coefficients

We only take the luminance blocks for information hiding. First of all, human eyes are relatively insensitive to the changes in brightness channel compared with the chroma channels. Second, modulating the nonzero integer transform coefficients in luminance block will not cause serious distortion to the quality of reconstructed image both subjectively and objectively. Third, nonzero coefficients are usually of low or medium frequencies, and changing them on small scale will not lead to a surge of code rate. Thus, the integer transform coefficients of luminance blocks are very suitable for taking on the role as the information hiding carriers. In addition, areas of the image that have rich texture often possess more complicated statistical property. It enables embedding secret information on such areas hard to perceive resulting in higher security. Therefore, we only choose quantization nonzero AC coefficients of specific I4-blocks in the macroblock that has rich texture as the carriers for information hiding.

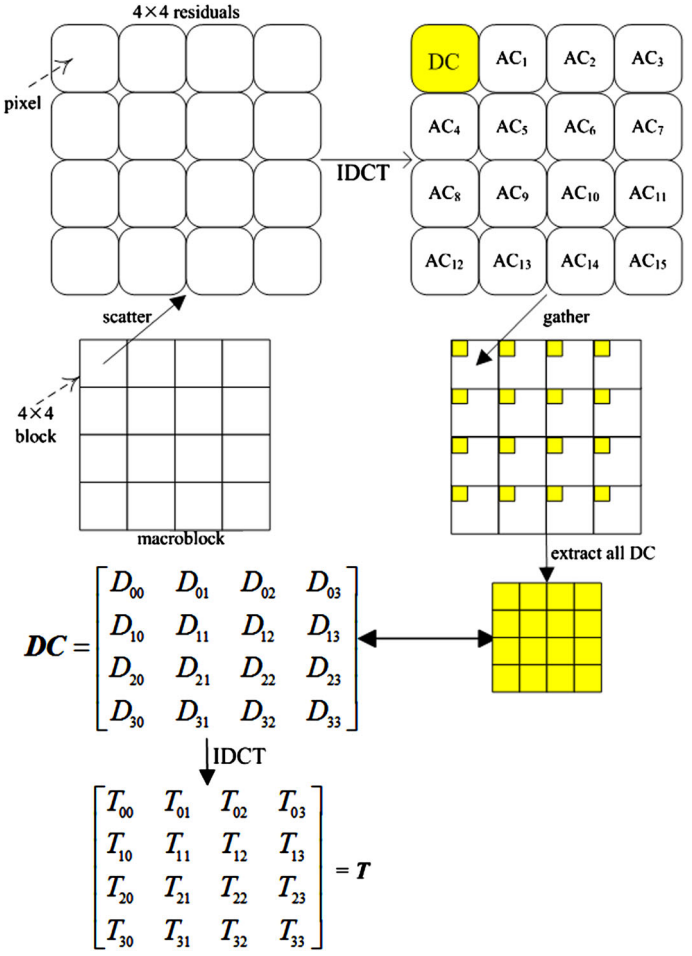

Fig. 3 Statistical process of quantitative macroblock texture complexity

First, we propose an approach to picking the macroblock that has rich texture in video frames according to the quantitative texture complexity. If we perform the integer transform on the 4×4 residual block of I4-block in the luminance channel macroblock, we will obtain a 4×4 matrix which contains 16 integer transform coefficients as shown in Fig. 3.

In Fig. 3, after IDCT, the upper left corner element is the DC component, while others are the AC components. We extract all the 16 DC components in a macroblock to form a DC matrix and then convert it into matrix T as shown in Fig. 3 using the integer transform again. For the macroblock that has rich texture, the number of nonzero AC coefficients of T is usually large and their values are large at the same time, according to the statistics. So we define the function $ACM(T)$ in (11) to quantify the average texture complexity of a macroblock. In (11), the symbols θ_1 and θ_2 are constant thresholds which are previously obtained by statistics on a large test set of

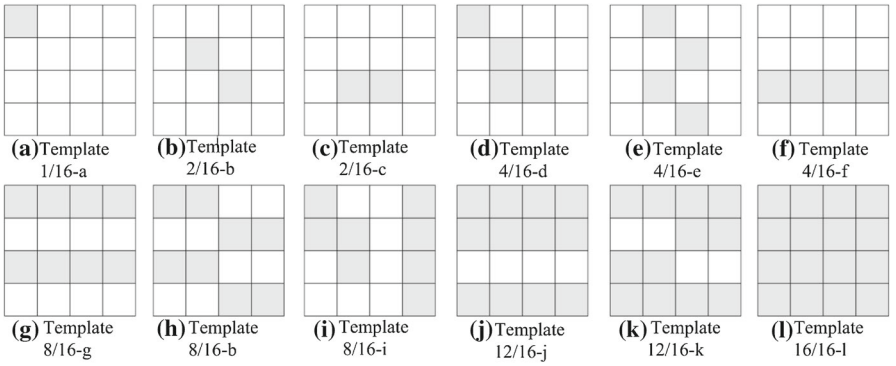

Fig. 4 Examples of EPT

video sequences. Only when the $ACM(T)$ is larger than θ_2, the macroblock will be taken as the host macroblock for next information embedding.

$$ACM(T) = \frac{\sum_{i=0}^{3}\sum_{j=0}^{3}\left[\left(|T_{ij}| > \theta_1\right)?\,|T_{ij}| : 0\right] - T_{00}}{\sum_{i=0}^{3}\sum_{j=0}^{3}\left[\left(|T_{ij}| > \theta_1\right)?1 : 0\right]} > \theta_2 \tag{11}$$

In order to reduce the impact on the decoding video quality and boost the security, we do not embed information in every I4-block in the selected macroblock. In fact, we only select certain I4-blocks as carrier according to the EPT and user's private key. The EPT determines not only as to which of the I4-blocks will be used for information hiding in a macroblock, but also the embedding density as shown in Fig. 4.

In Fig. 4, each small square represents the corresponding position of the I4-block in the macroblock and the shadowed squares represent the I4-blocks which will be used as carrier blocks according to a given EPT. In Fig. 4, we only give several EPT examples at different embedding density of 1/16, 1/8, 1/4 and so on. Actually, we can design plenty of templates by changing the embedding position and density. The upper bound for the number of different EPTs can be computed as follows:

$$C_{16}^{1} + C_{16}^{2} + C_{16}^{3} + \cdots + C_{16}^{16} = 16 + 120 + 560 + \cdots + 1 = 65535 \tag{12}$$

In this paper, we design all the 65,535 templates with a view to enhance the security of the hiding algorithm.

After obtaining the EPTs, we can dynamically choose the embedding templates according to a pseudo-random number sequence. We assume that the user in the information hiding side has a private key E_{k}. And using E_{k} as the seed of the pseudo-random number generator, we can get a pseudo-random number sequence x_k ($0 < x_k < 1,\ k = 1, 2, 3, $). Then, we perform integer quantification on x_k according to the following formula:

$$y_k = \lfloor x_k \times 65{,}535 \rfloor \tag{13}$$

We can use each y_k ($k = 1, 2, 3,$) in the sequence for determining as to which EPT will be selected for the current macroblock. We choose the EPT whose identification number is equal to y_k.

Through the above two steps, we can obtain a sequence of carrier I4-blocks: $B_1 \quad B_2 \quad B_3 \ldots B_m$ from a macroblock, where B_i ($i = 0, 1, \ldots$ m) denotes the coefficient matrix corresponding to the ith selected I4-block. Each matrix has 16 coefficients, so all the carrier coefficients can be arranged in one matrix as below:

$$B = \left[\begin{array}{cccc|cccc|c|cccc} b_{10} & b_{11} & b_{12} & b_{13} & b_{20} & b_{21} & b_{22} & b_{23} & & b_{m0} & b_{m1} & b_{m2} & b_{m3} \\ b_{14} & b_{15} & b_{16} & b_{17} & b_{24} & b_{25} & b_{26} & b_{27} & & b_{m4} & b_{m5} & b_{m6} & b_{m7} \\ b_{18} & b_{19} & b_{110} & b_{111} & b_{28} & b_{29} & b_{210} & b_{211} & \cdots & b_{m8} & b_{m9} & b_{m10} & b_{m11} \\ b_{112} & b_{113} & b_{114} & b_{115} & b_{212} & b_{213} & b_{214} & b_{215} & & b_{m12} & b_{m13} & b_{m14} & b_{m15} \end{array}\right] \quad (14)$$

We will select the nonzero coefficients from B according to the following algorithm:

Algorithm 1 Select Nonzero coefficients from B

Input : B
Output : *NZCPL* (ListStored Nonzero AC Coefficients Position)
1 : Clear *NZCPL*
2 : while true **do**
3 : Get Coefficient c from B in zig-zag order
4 : if c is NULL **then**
5 : Jump out of the loop
6 : else if c is not DC **and** c is Nonzero **then**
7 : Add Position of c to *NZCPL*
8 : end if
9 : end while

Through the above process, we can select all the nonzero AC coefficients from the macroblock and obtain a nonzero coefficient sequence $(c_1, c_2, \ldots, c_z)$, which will be used in the next information embedding step.

3.2 Embedding Information by Modulating AC Coefficients

We first establish an N-dimensional virtual space (NVS). We take every N host AC coefficients as a group and map it into certain element in the NVS, by converting the N coefficients into the N-dimensional coordinates of NVS according to our mapping rules. Then, with the help of NVS, we can effectively implement the embedding process. Next, we will elaborate this process in detail.

Firstly, we take every $N (N \leqslant Z)$ elements sequentially from the nonzero coefficient sequence $(c_1, c_2, \ldots, c_z)$ to form a coordinate set $C = (c_1, c_2, \ldots, c_N)$, where the ith dimensional coordinate c_i is an integer. Its value range is $[-127, 127]$ according to the definition of H.264/AVC integer transform. The value of each space lattice in the NVS is assigned according to the following function F:

$$F(C) = F(c_1, c_2, \ldots, c_N) = (y(c_1) + 2y(c_2) + \cdots + Ny(c_N)) \mathrm{Mod}(2N + 1) \quad (15)$$

where the value of $y(c_i)$ is the ith dimensional coordinate of the NVS. Its value is determined by the following formula:

$$y(c_i) = \begin{cases} c_i \text{Mod}(2N+1) & c_i \geq 0 \\ [(c_i+1)\text{Mod}(2N+1) + 2N + 1]\text{Mod}(2N+1) & c_i < 0 \end{cases} \tag{16}$$

In this way, the NVS is established. We call the above process as the virtual space encoding (VSE) in this paper. In fact, we map the carrier coefficient set C into a $2N+1$-ary number by the VSE process. An example of the NVS with $N = 3$ is show in Fig. 5. After the VSE process, we can use the NVS for embedding. For each space lattice of the NVS, if its value is equal to the to-be-embedded $2N+1$-ary number, no coefficient needs to be changed; otherwise, certain coefficient in the group will be changed to +1 or -1 to match the values.

If we have X bits secret data to hide, we need to convert them into the $2N+1$-ary number sequence first. And then, these $2N+1$-ary numbers will be embedded one after another. Obviously, to embed one $2N+1$-ary number, we need N carrier coefficients. Suppose that D_i is the current to-be-embedded $2N+1$-ary number and C_i is the current carrier set, then the embedding process is carried out using the following three steps. First, the value of $F(C_i) = F(c_1, c_2, \ldots, c_k, \ldots, c_N)$ will be computed according to (15). Second, comparing the value of $F(C_i)$ and D_i, if they are equal, then the embedding process is terminated early. Or else, we will go to step 3. In this step, we first take $c_k (k = 1, 2, \ldots, N)$ from C_i sequentially and perform a subtraction operation on c_k according to the following formula:

$$c'_k = \begin{cases} c_k - 1 & \text{if } c_k \neq 1 \\ -1 & \text{if } c_k = 1 \end{cases} \tag{17}$$

where c'_k is the modified c_k; if $F(c_1, c_2, \ldots, c'_k, \ldots, c_N) = D_i$, then we can terminate the hiding process. Or else, we still need to perform the addition operation on c_k according to the following formula:

$$c_k = c'_k = \begin{cases} c_k + 1 & \text{if } c_k \neq -1 \\ 1 & \text{if } c_k = -1 \end{cases} \tag{18}$$

If $F(c_1, c_2, \ldots, c'_k, \ldots, c_N) = D_i$, then we can end the hiding process at this point. Or else, we must take the next c_k and repeat the step 3 again. We can prove that there exists one and only one c'_k that can meet the condition: $F(c_1, c_2, \ldots, c'_k, \ldots, c_N) = D_i$. The proof follows.

First, according to (15) and (16), the value of any space lattice e in the NVS can be rewritten as:

$$\begin{aligned} F(W_1, W_2, \ldots, W_i, \ldots, W_N) &= (W_1 + 2W_2 + \cdots + iW_i + \cdots + NW_N) \\ &= K\text{Mod}(2N+1) \end{aligned} \tag{19}$$

where $W_i (i = 1, 2, \ldots, N)$ is the ith dimensional coordinate of the NVS which belongs to set $\{0, 1, \ldots, 2N\}$ and K is an integer. Then, the values of the two adjacent

lattices of e in ith dimension will be:

$$\begin{aligned} &F(W_1, W_2, \ldots, W_i - 1, \ldots, W_N) \\ &= (W_1 + 2W_2 + \cdots + i(W_i - 1) + \cdots + NW_N)\text{Mod}(2N + 1) \\ &= (W_1 + 2W_2 + \cdots + iW_i + \cdots + NW_N - i)\text{Mod}(2N + 1) \\ &= (K - i)\text{Mod}(2N + 1) \end{aligned} \tag{20}$$

and:

$$\begin{aligned} &F(W_1, W_2, \ldots, W_i + 1, \cdots, W_N) \\ &= (W_1 + 2W_2 + \cdots + i(W_i + 1) + \cdots + NW_N)\text{Mod}(2N + 1) \\ &= (W_1 + 2W_2 + \cdots + iW_i + \cdots + NW_N + i)\text{Mod}(2N + 1) \\ &= (K + i)\text{Mod}(2N + 1) \end{aligned} \tag{21}$$

Thus, by varying i from 1 to N, we can obtain the set of values of e and its neighbours as: $V = \{ (K - N) \text{ Mod } (2N + 1), \ldots, (K - 1) \text{ Mod } (2N + 1), K \text{ Mod } (2N + 1), (K + 1) \text{ Mod } (2N + 1), \ldots, (K + N) \text{ Mod } (2N + 1) \}$. Obviously, the values of e and its neighbours are $2N + 1$-ary numbers and different from one another. Furthermore, because K can be any integer, we can assume its value is N, so the set V can be transformed to: $V = \{ 0, \ldots, N-2, N-1, N, N+1, N+2, \ldots, 2N \}$ which can be rewritten as set $\varphi = \{ 0, 1, 2, \ldots, N, \cdots, 2N \}$. Obviously, if D_i is not equal to the value of e, then we can surely find a neighbour of e whose value is D_i, because D_i is a $2N + 1$-ary number and belongs to set φ.

For ease of understanding, let us consider embedding 7-ary numbers as an example as shown in Fig. 5. First, the secret bits are converted into the 7-ary number sequence. Assume that we will embed the first 7-ary number 2 in the sequence now. The first step is to take three selected host coefficients to form the set coefficient $C = (30, 38, 15)$ and then translate the decimal number coefficients into 7-ary number according to (16). Then, we can get the coordinate set $C = (2, 3, 1)$ of the NVS and its mapped 7-ary number is $F(2, 3, 1) = (2+2\times3+3\times1)$ Mod 7=4. Because the to-be-embedded 7-ary number is 2, we need to change one coefficient in the set C. We can find that changing the second coefficient 38 into 37 can satisfy $F(C) = 2$ according to the embedding process step 3 described above. Using similar operations, we can embed other 7-ary numbers in the sequence. In Fig. 5, we give a schematic diagram of the complete embedding process when the dimension of the NVS is 3.

In conclusion, if the value of the mapped element of the NVS is equal to the to-be-embedded number, then we do not need to modulate the coordinate set composed of the integer transform coefficients. Otherwise, we need to modify one and only one coefficient to form the new coordinate set which would be mapped into a new element of the NVS whose value is equal to the to-be-embedded number. Obviously, our method can effectively improve the embedding efficiency compared with current ITCM methods. The proposed embedding approach is able to achieve the performance of embedding $\text{Log}_2(2N + 1)$ bits by modifying one coefficient, while the current methods just embed one bit only. Specifically, the coefficient modification of our method is very trivial, only adding or subtracting 1 to a certain coefficient.

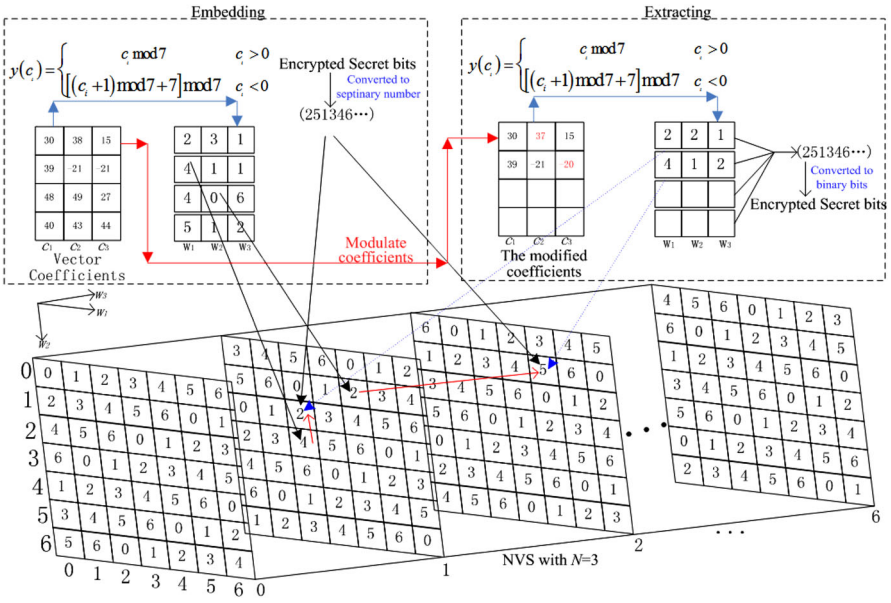

Fig. 5 Schematic diagram of information embedding and extracting based on N-dimensional virtual space with $N = 3$

4 Extracting Secret Bits from H.264/AVC Stream

The secret bits can be extracted blindly. We assume that the user's private key in the encoding side is transmitted in other security channel. In this case, we only need the integer transform coefficients to restore the secret bits. And the integer transform coefficients can be obtained by partially decoding the H.264/AVC compressed video bit stream. After decoding, we can determine as to which of the coefficients are selected as carriers according to the process defined in the left part of Fig. 2. The first step is to select the host macroblock based on its texture diversity and then choose the EPT using the private key as the seed of the pseudo-random number generator. Finally, using the nonzero AC coefficient selecting algorithm given in Sect. 3, we can obtain the carrier coefficient sequence $(c_1, c_2, \ldots, c_z)$, which will be used in the next information extracting step.

Similar to the embedding process, we also take every $N (N \leqslant Z)$ elements sequentially from the nonzero coefficients sequence $(c_1, c_2, \ldots, c_z)$ to form a coordinate set $C = (c_1, c_2, \ldots, c_N)$ of the NVS. Then, we can obtain one $2N + 1$-ary number, which is exactly the value of the space lattice at coordinate set C as follows. First, we compute the intermediate value $y(c_i)$ according to (16) for every c_i $(i = 1, 2, \ldots, \mathrm{N})$. After obtaining all the $y(c_i)$ values, the hidden $2N + 1$-ary number D can be directly computed according to the following formula:

$$D = (y(c_1) + 2y(c_2) + \cdots + Ny(c_N))\mathrm{Mod}(2N + 1) \tag{22}$$

Let us also consider extracting 7-ary numbers for example as shown in Fig. 5. First let us see how to extract the 7-ary number 2 embedded in the encoding side. The first step is to decode the H.264/AVC stream and take the first three host coefficients to form the coefficients set $C = (30, 37, 15)$ and then translate the decimal number coefficients into the 7-ary number according to (16). Next, we can get the coordinate set $C = (2, 2, 1)$ of the NVS and its mapped 7-ary number is $F(2, 2, 1) = (2 + 2 \times 2 + 3 \times 1)$ Mod 7=2. In this way, we can easily get the hidden 7-ary number 2 in the decoding side. In Fig. 5, we also give the schematic diagram of the blind extracting process when the dimension of the NVS is 3.

Finally, we can transform all the extracted $2N+1$-ary numbers to binary bits, which are the hidden encrypted secret bits. After decrypting them by using the user's private key, we can get the complete secret information in the video encoding side.

Obviously, the retrieval of the hidden secret information is simple and fast. It only requires that the integer transform coefficients be accomplished without the original media or complete decoding. This merit makes the extracting process of our method very suitable for practical application.

5 Experiment Results and Discussion

In this section, we evaluate the performance of the proposed method in terms of several aspects. First, we investigate the information hiding impact on the decoding video image quality on the basis of both the subjective impression and the objective quantitative assessment. And then, we will observe the increment of the compressed video code rate caused by the data hiding. Furthermore, we also evaluate the counteracting steganalysis performance of our method. We also give the experiment results in all these three aspects for the methods in [2,4,10,12,16,23,28,32] under the same conditions for comparison purpose. We realize our approach and the methods in [2,4,10,12,16,23,28,32] based on the H.264/AVC reference software version JM16.0. The tests are performed using H.264 Main Profile configuration (with RDO, CABAC, QP=25, and 15 frames/sec) and with a GOP structure of "IBPIBPIBP". All the experiments are carried out under the modified JM model with the above settings using the randomly selected standard test QCIF sequences: Mobile, Coastguard, Container and Foreman.

5.1 Decoding Video Quality

Video quality includes subjective impression and objective assessment. Subjective impression mainly emphasizes the intuitive perception of the video such as whether there exists severe distortion or not through human eyes. Meanwhile, objective assessment is a quantitative method of video quality and has multiple different standards. Since the peak signal-to-noise ratio (PSNR) can effectively quantify the video image quality, it is widely used in the objective assessment of video quality. For this paper, we also adopt the PSNR as the indicator of the quality of the decoding video which contains the hidden information.

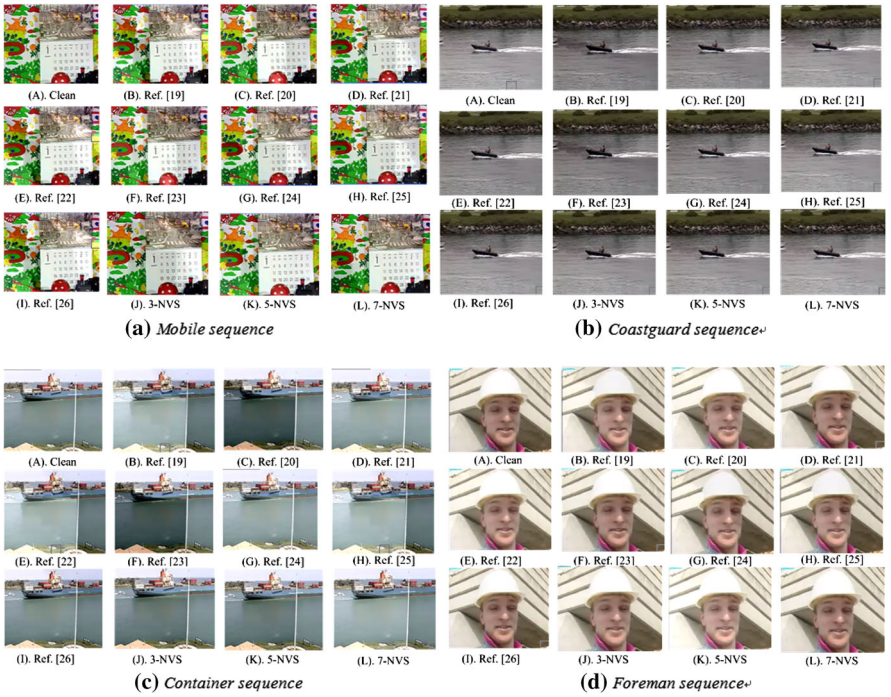

Fig. 6 Decoding video image before and after information embedding using different steganography methods

First, we give some example images for intuitive perception of the watermark embedding impact, as shown in Fig. 6. In this figure, we can see some randomly selected images from the decoding video of the Mobile, Coastguard, Container and Foreman test video sequences, respectively. For direct comparison, the images from the same original test video sequence but under different encoding strategies are put in a subfigure. There are several encoding strategies: encoding without any information hiding (clean), encoding with secret information embedding using the methods in references [2,4,10,12,16,23,28,32] and encoding with secret information embedding through our method with different dimensions of the NVS. Our method improves the visual effect compared with that of the methods in [2,4,10,12,16,23,28,32] when the dimension of NVS is large, although there is difficulty in seeing this in Fig. 6 due to the figure being too small. However, this conclusion can be further confirmed by the quantitative evaluation based on PSNR.

Next, we give the objective performance evaluation results. Table 1 presents the changes in the average luminance PSNR value of the four selected test sequences after embedding the same amount of secret bits. In the table, the first eight rows refer to the objective video quality after the information hiding using the methods in [2,4,10,12,16,23,28,32], and the "n-NVS" ($n = 3, 5, 7$) rows refer to the results using our methods with different dimensions of the NVS. Obviously, our method's average PSNR values of the four test sequences are all larger than those of the methods

Table 1 Average PSNR of decoding video images of different steganography methods

	Coastguard	Container	Foreman	Mobile
Ref. [18]	29.16	28.32	30.95	25.58
Ref. [19]	29.11	28.32	30.82	25.47
Ref. [20]	29.23	28.41	31.13	25.68
Ref. [21]	29.27	28.38	31.12	25.46
Ref. [22]	29.18	28.39	30.69	25.11
Ref. [23]	29.21	28.35	31.43	26.01
Ref. [24]	28.47	27.76	29.41	24.21
Ref. [26]	29.39	28.47	32.34	26.33
3-NVS	29.33	29.51	32.01	27.16
5-NVS	30.15	30.43	32.69	27.33
7-NVS	30.67	28.96	33.43	27.28

in [2,4,10,12,16,23,28,32]. And the most improvement is about 8 % for the Foreman sequence. Therefore, our method has less impact on the decoding video quality on the basis of both the subjective and objective standards, when compared with other current state-of-the-art methods.

5.2 Increment of Code Rate

Generally, the code bit rate of compressed video will increase once the video is processed with information hiding, because the most optimal integer DCT coefficients can be changed. However, the extent of code rate increase in different information hiding methods is different too. And the better method ought to boost the bit rate as little as possible when embedding the same amount of secret data. So we evaluate the information embedding impact on code rate of our method and compare it with those of the methods in [2,4,10,12,16,23,28,32] too. For each test sequence, we encode 100 frames and embed the same amount of secret bits using the methods in [2,4,10,12,16,23,28,32] and our method with different dimensions of the NVS, respectively. The coding bits of different frames are shown in Fig. 7. As seen from Fig. 7, the coding bits of each frame in our method decrease compared to the methods in [2,4,10,12,16,23,28,32], especially when the dimension of the NVS is large. However, this conclusion is not entirely intuitive from the figure due to the space limitation. So, with a view to confirm the conclusion, we also give specific statistics results of code rate of each sequence in Table 2.

We see that the code rates of our method with different dimensions are all smaller than that of any other methods in [2,4,10,12,16,23,28,32] from this table. Taking the Mobile sequence for example, the code rate of Ref. [10] is about 1705.44 Kbps, and the code rate of our method is about 1699.68 Kbps, when the dimension of the NVS is 3. Moreover, the code rate will decrease as the dimension increases, e.g. the code rate reduces to 1698.38 Kbps when the dimension of the NVS increases to 7. For other test sequences, we can observe and come to the same conclusion. So it is safe to say that our method has an improvement from the point of view of code rate, compared

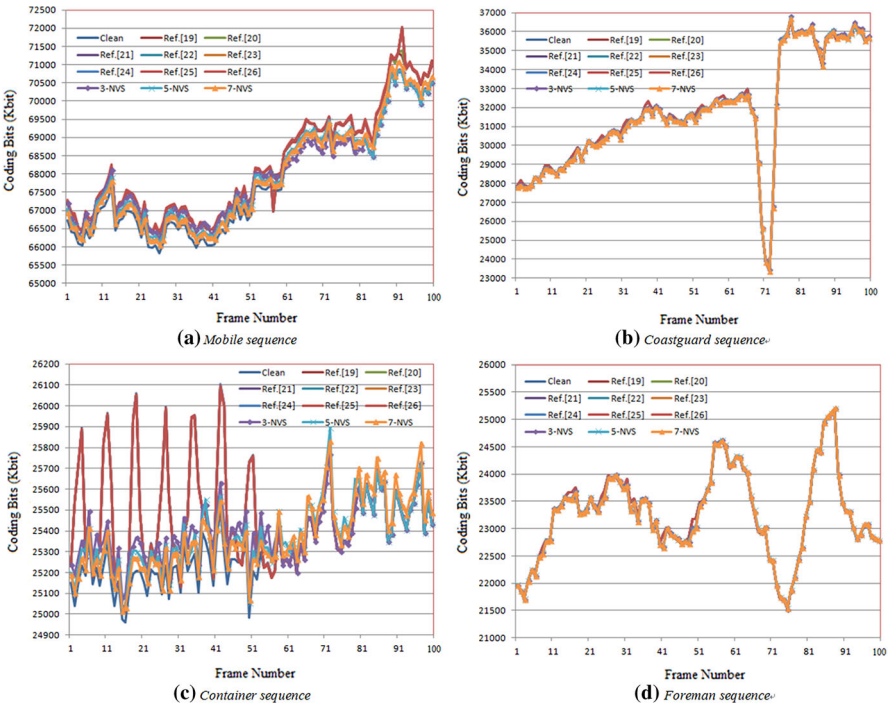

Fig. 7 Coding bits of different frames before and after information hiding using different steganography methods

Table 2 Code rate (KBPS) of compressed video of different steganography methods

	Coastguard	Container	Foreman	Mobile
Ref. [18]	794.77	636.68	582.22	1705.44
Ref. [19]	794.81	636.69	582.29	1705.59
Ref. [20]	794.81	636.67	582.19	1705.41
Ref. [21]	794.70	636.75	582.21	1705.41
Ref. [22]	794.74	636.44	582.08	1705.39
Ref. [23]	794.69	635.99	582.34	1704.91
Ref. [24]	794.56	636.41	582.17	1704.69
Ref. [26]	799.18	637.70	583.49	1709.14
3-NVS	794.41	634.79	581.94	1699.68
5-NVS	794.01	634.48	581.89	1698.95
7-NVS	793.85	634.40	581.86	1698.38

with methods in [2,4,10,12,16,23,28,32]. And generally, this trend will be enhanced as the dimension of the NVS increases. The reason is that the embedding efficiency of our method increases as the dimension of the NVS increases.

We define the embedding efficiency E as the average number of hidden secret bits per host coefficient (modified or not). Because our approach can embed one $2N + 1$-ary number when a coefficient is changed or not, the embedding efficiency is the bit

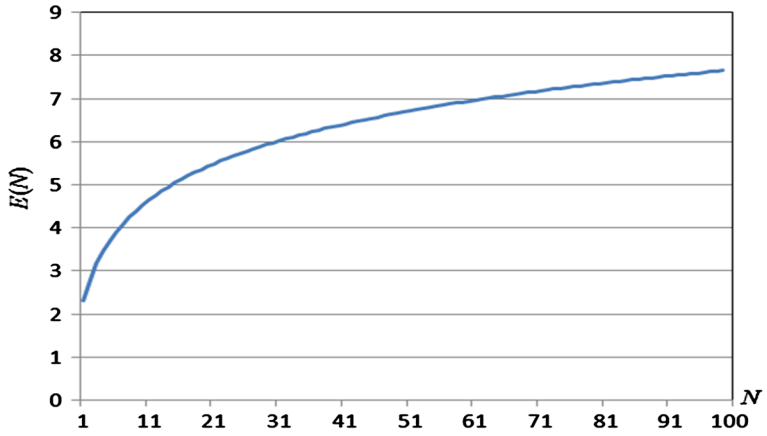

Fig. 8 $E(N)$ curve along with the change in N

numbers of a $2N+1$-ary number, which can be computed as follows:

$$E(N) = \mathrm{Log}_2(2N+1) \tag{23}$$

The changing trend of embedding efficiency along with the variation of $N(N>1)$ is shown in Fig. 8.

We can see that our method's embedding efficiency is always larger than 2 which means that our method will embed two bits at least when one coefficient is modified. Furthermore, the embedding efficiency will increase to infinity as N increases. Meanwhile, the embedding efficiency E of current information hiding methods based on H.264/AVC integer transform coefficients modulation [2,4,10,12,16,23,28,32] is always the same; for these methods, E is always equal to one, which means that modifying one coefficient will only embed one secret bit. So our method has a better embedding efficiency. And with the increase in the embedding efficiency, the concealment of the system will improve when given the same embedding capacity, because the number of coefficients that needed to be modified reduce. For example, if we have X bits secret information to hide, the current methods will need to modify X coefficients at most, while our approach only needs to modify $X/\mathrm{Log}_2(2N+1)$ coefficients (which is smaller than $X/2$) when N is larger than 1. This is why the code rate of our method decreases when the dimension of NVS increases.

5.3 Counteracting Steganalysis Performance

If the information hiding can be detected with a high probability, then we can say that it is not a good information hiding method. Thus, the counteracting steganalysis performance is an important evaluation criterion of an information hiding method. So we evaluate the counteracting steganalysis performance of our method and compare it with those of the methods in [2,4,10,12,16,23,28,32].

Table 3 Accuracy of information hiding detection of different steganography methods

	Coastguard (%)	Container (%)	Foreman (%)	Mobile (%)
Ref. [18]	51.00	51.75	51.00	51.25
Ref. [19]	51.25	51.75	52.25	52.25
Ref. [20]	50.00	51.00	52.00	51.00
Ref. [21]	51.25	52.25	51.75	51.25
Ref. [22]	51.00	52.50	52.00	52.00
Ref. [23]	50.75	52.25	52.25	51.75
Ref. [24]	50.75	52.00	51.00	51.75
Ref. [26]	52.25	53.00	51.00	52.75
3-NVS	50.00	50.50	50.00	50.00
5-NVS	50.00	50.00	49.25	50.00
7-NVS	49.75	50.25	49.00	49.25

The steganalysis method in [25] models the differences between adjacent pixels using first-order and second-order Markov chains. Subsets of sample transition probability matrices are then used as features for a steganalyser implemented by the support vector machine. It has superior accuracy for detecting steganography in the transform domain. So, in this paper, we adopt this method for evaluating the counteracting steganalysis performance of our method and the methods in [2,4,10,12,16,23,28,32]. For each test sequence, we first encode and decode it to get the "cover" sample frames, and then, we encode and decode it again using different information hiding method, respectively, to get the "stego" sample frames. After generating the training and test datasets, we build the steganalyser proposed in [25] to determine whether there exists information hiding in the decoding video frames or not. The experimental results are shown in Table 3. We can see that the detection accuracy of our method is lower than those of the methods in [2,4,10,12,16,23,28,32]. Thus, our method has superior counteracting steganalysis performance when compared with other methods.

In conclusion, we see from our experiments that our method has some advantages in the all three aspects. The reasons can be summarized as follows. First, the watermark bits are only embedded into the specially selected macroblocks in our method. Only the macroblocks that have rich texture are chosen as host macroblocks. In such macroblocks, the integer DCT coefficients are often larger and slight changes are hard to perceive. And I4-blocks in the host macroblock being used for information embedding are various. Therefore, our method has higher transparency. Moreover, the introduction of mapping rules based on the NVS makes the change in integer DCT coefficient as little as possible, which further enhances the concealment. Second, the secret bits are hidden through a slight change in integer DCT coefficients (by just +1 or −1), and no additional information is added to the original video stream. This will benefit to keep statistical characteristics of the integer DCT coefficients and prevent steganalysis attacks. Third, the number of integer DCT coefficients is very huge in the H.264/AVC compressed stream, and in general, only a small part will be taken as carrier for secret information under given embedding capacity using our method; hence, the change in the compressed video caused by hiding will stay at a low level. Therefore, similar to

adding a spoonful of blue ink to water in a big swimming pool, it is too hard to detect. Furthermore, we define enough EPTs with different capacities and grouping criteria, and the selection of the EPT is controlled by the user's private secret key. Therefore, there are multiple templates and the selection of a certain template is determined by the private key of a user. These factors can effectively guarantee the security of our method even when the attacker masters the implementation details of our method.

6 Conclusion

We have proposed a novel highly efficient H.264/AVC information hiding method in this paper. We can embed a message of $2N + 1$ values into a group of N coefficients by changing no or only one coefficient in the group since each change has two choices. Additionally, in order to increase the transparency and security of the algorithm as well as to reduce its damage to the decoding video quality, this paper also has introduced a host macroblock selection mechanism based on image texture complexity and defined embedding templates to choose the host I4-blocks in the selected macroblock. Through experiments, we establish that our approach has excellent performance when compared with other integer DCT coefficients modulation methods. In the future, we will try to extend our method to the latest video encoding standard H.265/HEVC.

Acknowledgments This work is supported partly by Natural Science Foundation of Nation and Hainan Province of China under Grant 61303249 and 614236, partly by Important Science and Technology Project of Hainan Province of China under Grant JDJS2013006 and partly by Preferred Foundation of Director of Institute of Acoustics, Chinese Academy of Sciences. The authors would like to thank the Editor-in-chief, Prof. M. N. S. Swamy, for helping us to improve the presentation of the paper.

References

1. M.A. Ali, E.A. Edirisinghe, Multi-layer watermarking of H.264/AVC video using Differential Expansion on IPCM blocks. In: 2011 IEEE International Conference on Consumer Electronics (ICCE), pp. 53–54 (2011)
2. S.K. Bavipati, X. Su, Secure compressed domain watermarking for H.264 Video. In: 2010 Seventh International Conference on Information Technology: New Generations (ITNG), pp. 387–391 (2010)
3. H. Bin, Z. Li-Yi, Z. Wei-Dong, A novel steganography algorithm based on motion vector and matrix encoding. In: 2011 IEEE 3rd International Conference on Communication Software and Networks (ICCSN), pp. 406–409 (2011)
4. M. Fallahpour, M. Semsarzadeh, S. Shirmohammadi, J. Zhao, A realtime spatio-temporal watermarking scheme for H.264/AVC. In: 2013 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), pp. 872–875 (2013)
5. G. Feng, G.Z. Wu, Motion vector and mode selection based fragile video watermarking algorithm. In: 2011 IEEE International Conference on Anti-Counterfeiting, Security and Identification (ASID), pp. 73–76 (2011)
6. Y. Hu, C. Zhang, Y. Su, Information hiding based on intra prediction modes for H.264/AVC. In: IEEE International Conference on Multimedia and Expo, pp. 1231–1234 (2007)
7. S.K. Kapotas, A.N. Skodras, Real time data hiding by exploiting the IPCM macroblocks in H.264/AVC streams. J. Real-Time Image Process. **4**(1), 33–41 (2009)
8. S.K. Kapotas, E. Varsaki, A.N. Skodras, Data hiding in H.264 encoded video sequences. In: IEEE 9th Workshop on Multimedia Signal Processing, MMSP 2007, pp. 373–376 (2007)

9. T.-Y. Kuo, Y.-C. Lo, C.-I. Lin, Fragile video watermarking technique by motion field embedding with rate-distortion minimization. In: International Conference on Intelligent Information Hiding and Multimedia Signal Processing, pp. 853–856 (2008)
10. Y. Li, H. Chen, Y. Zhao, A new method of data hiding based on H.264 encoded video sequences. In: 2010 IEEE 10th International Conference on Signal Processing (ICSP), pp. 1833–1836 (2010)
11. K. Liao, D. Ye, S. Lian, Z. Guo, J. Wang, Lightweight information hiding in H.264/AVC video stream. In: International Conference on Multimedia Information Networking and Security, MINES'09, 1, pp. 578–582 (2009)
12. A. Mansouri, A.M. Aznaveh, F. Torkamani-Azar, F. Kurugollu, A low complexity video watermarking in H.264 compressed domain. IEEE Trans. Inf. Forensics Secur. **5**(4), 649–657 (2010)
13. N. Mehmood, M. Mushtaq, *Blind Watermarking Scheme for H. 264/AVC Based on Intra 4x4 Prediction Modes. Future Information Technology, Application, and Service* (Springer, Amsterdam, 2012)
14. B.G. Mobasseri, Y.N. Raikar, Authentication of H.264 streams by direct watermarking of CAVLC blocks. Electronic Imaging 2007, International Society for Optics and Photonics, 65051W–65051W-5 (2007)
15. M. Najla, F. Omid, H.264 Copyright protection with motion vector watermarking. In: International Conference on Audio, Language and Inamge Processing, pp. 1384–1389 (2008)
16. M. Noorkami, R.M. Mersereau, Compressed-domain video watermarking for H.264. In: IEEE International Conference on Image Processing, ICIP, 2, pp. II-890–893 (2005)
17. W. Pei, Z. Zhendong, L. Li, A video watermarking scheme based on motion vectors and mode selection. In: 2008 International Conference on Computer Science and Software Engineering, 5, pp. 233–237 (2008)
18. G.J. Simmons, The subliminal channels of the US Digital Signature Algorithm. In: Proceedings of the 3rd Symposium on the State and Progress of research in Cryptography, pp. 35–54 (1993)
19. G.J. Simmons, The prisoners problem and the subliminal channel. In: Proceedings of Crypto Advances in Cryptology, pp. 51–67 (1984)
20. G.J. Simmons, Subliminal channels; past and present. Eur. Trans. Telecommun. **5**(4), 459–474 (1994)
21. K. Swaraja, Y.M. Latha, V. Reddy, A. Paramkusam, video watermarking based on motion vectors of H.264. In: Annual IEEE India Conference (INDICON), pp. 1–4 (2011)
22. Y. Tew, K. Wong, An overview of information hiding in H.264/AVC compressed video. IEEE Trans. Circuits Syst Video Technol. **24**(2), 305–319 (2014)
23. L. Tian, N. Zheng, J. Xue, A blind and spatial–temporal based video watermarking for H. 264/AVC. In: 2011 First Asian Conference on Pattern Recognition (ACPR), pp. 598–602 (2011)
24. L. Tian, N. Zheng, J. Xue, T. Xu, A CAVLC-based blind watermarking method for H.264/AVC compressed video. In: IEEE Asia-Pacific Services Computing Conference, APSCC'08, pp. 1295–1299 (2008)
25. P. Tomás, B. Patrick, F. Jessica, Steganalysis by subtractive pixel adjacency matrix. IEEE Trans. Inf. forensics Secur. **5**(2), 215–224 (2010)
26. T. Wiegand, G.J. Sullivan, G. Bjontegaard, A. Luthra, Overview of the H. 264/AVC video coding standard. IEEE Trans. Circuits Syst. Video Technol. **13**(7), 560–576 (2003)
27. D. Xu, R. Wang, Watermarking in H.264/AVC compressed domain using Exp-Golomb code words mapping. Opt. Eng. **50**(9), 097402–097402-11 (2011)
28. D. Xu, R. Wang, J. Wang, Blind digital watermarking of low bit-rate advanced H.264/AVC compressed video. Digital Watermarking (Springer, Berlin, 2009) pp. 96–109
29. D. Xu, R. Wang, J. Wang, Prediction mode modulated data-hiding algorithm for H.264/AVC. J. Real-Time Image Process. **7**(4), 205–214 (2012)
30. D. Xu, R. Wang, J. Wang, Low complexity video watermarking algorithm by exploiting CAVLC in H. 264/AVC. In: 2010 IEEE International Conference on Wireless Communications, Networking and Information Security (WCNIS), pp. 411–415 (2010)
31. G. Yang, J. Li, Y. He, Z. Kang, An information hiding algorithm based on intra-prediction modes and matrix coding for H.264/AVC video stream. AEU Int. J. Electron. Commun. **65**(4), 331–337 (2011)
32. L. Zhang, Y. Zhu, L.M. Po, A novel watermarking scheme with compensation in bit-stream domain for H. 264/AVC. In: 2010 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), pp. 1758–1761 (2010)
33. D. Zou, J.A. Bloom, H. 264 stream replacement watermarking with CABAC encoding. In: 2010 IEEE International Conference on Multimedia and Expo (ICME), pp. 117–121 (2010)