Multimedia Tools and Applications (2019) 78:8167–8181
https://doi.org/10.1007/s11042-018-6729-3

Two-dimensional histogram modification based reversible data hiding using motion vector for H.264

Dong Li[1] · YingNan Zhang[1] · XinChao Li[1] · Ke Niu[1] · XiaoYuan Yang[1] · YuJuan Sun[2]

Received: 17 May 2018 / Revised: 24 July 2018 / Accepted: 25 September 2018 /
Published online: 3 October 2018

Abstract
Histogram modification (HM) is an efficient algorithm for reversible data hiding into H.264 video, motion vector based methods have been applied to HM. In this paper, we present a special strategy of reversible data hiding using motion vector based on two-dimensional histogram modification. Firstly, the two components of a motion vector will compose an embedding pair. The values of embedding pairs are classified into 17 non-intersect sets. Then, according to the set that the embedding pair belongs to, we can embed data into motion vectors by modifying the value of embedding pair. Experimental results verify that the embedding capacity is increased by two times compared with the previous algorithm, and the PSNR declines about 5%. The reversibility of the algorithm is also studied, three rules are presented to judge the reversibility of an algorithm directly, and the rules are suitable for other histogram modification based methods.

Keywords Reversible data hiding · H.264/AVC · Histogram modification · Motion vector · Two-dimensional

1 Introduction

Reversible data hiding (RDH) means the original medium can be completely recovered after the hidden data has been extracted [1]. For watermark, robustness is more important, Zhang et al. [18] proposed a robust image steganography that the message can be correctly transmitted even after JPEG compression, but in the fields such as medical, military and legal ones, the distortion of images and videos is sometimes unacceptable, that is, the irreversible loss of carrier is not allowed [2]. RDH can solve this problem properly and the information can be extracted with the carrier recovered completely. The first RDH method was proposed for

✉ XiaoYuan Yang
18792733969@163.com

[1] Key Laboratory of Network and Information Security under the People's Armed Police, Electronic Department, Engineering University of People's Armed Police, Xi'an 710086, China

[2] Department of Electrical and Computer Engineering University of Toronto, Toronto, ON, Canada

image authentication [3]. At present, difference expansion (DE) and histogram modification (HM) are two mainly algorithms used for RDH in images. Compared with image, videos have lots of advantages. Higher quality allows a larger distortion, and more redundancy means a higher embedding capacity. Since H.264 compression standard has become the mainstream video codec, it is necessary to find ways to apply RDH methods into H.264 video.

Tian proposed DE for RDH into images [13]. The difference between two pixels is extended to embed one bit with their average value keeps intact, and the process is reversible. A novel RDH algorithm based on two-dimensional histogram modification is proposed [19], in this method, two quantified DCT (QDCT) coefficients are randomly selected from a 4 × 4 block, the two coefficients will form a pair, according to the set that the pair belongs to, the value of the pair is modified to embed data, and the algorithm possesses a large embedding capacity.

HM algorithm has drawn attention of researchers. In order to decrease the distortion, our group introduces a parameter K stands for the reference frame interval [10]. By modifying K, a large embedding capacity and good visual quality can be both got. The data is embedded into the horizontal components or the vertical ones of motion vectors (MV) by modifying the histogram, however when the histogram is modified once, only one-bit data can be embedded, which limits the embedding capacity.

In this paper, we present a novel algorithm based on two-dimensional histogram modification of MVs. The vertical and horizontal component of a MV constitute an embedding pair, and the values of embedding pairs are classified into 17 non-intersect sets. By modifying the two components of a motion vector simultaneously, more data can be embedded at one time, and the distortion performance is acceptable.

2 Preliminaries

2.1 Motion compensation and motion vector

H.264 compression standard is efficient in video coding, which supports multiple blocks from 16 × 16 to 4 × 4. Under the 8 × 8 mode, the blocks can be divided into8 × 4, 4 × 8, 4 × 4 sub-blocks, as shown in Fig. 1 [11]. The blocks are estimated in 1/4-pixel precision, and it means H.264 has more MVs compared with previous compression standards such as MPEG-2 etc. Therefore, MV is an ideal carrier for data embedding. Lin et al. [6] proposed an algorithm that MVs are divided into different groups, and some of them are chosen to embed watermark. MVs are suitable for steganography as well. In Song's algorithm [12], researchers presented a

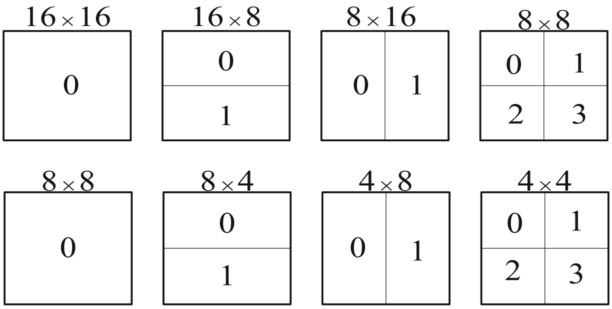


Fig. 1 Partition of blocks

reversible video steganography scheme to hide information into the motion vector of each block for 3D MVC videos. MVs can also be employed to achieve error concealment [15].

Discrete Cosine Transform(DCT) is another key step of H.264 standard. DCT employs an integer transformation based on 4×4 sub-blocks,DCT blocks become QDCT blocks after quantification, and many RDH algorithms are implemented on QDCT coefficients [7], in H.264/AVC, treat the to-be embedded data with BCH syndrome code, and then embed the encoded data into the QDCT coefficients, in this way, robustness can be improved.

2.2 Previous algorithm

H.264 possesses a higher prediction accuracy, it means the histogram of H.264 video is more smooth than previous standard, and there are more 1 and − 1 points in H.264’s motion vector histogram. Our group proposed an algorithm (MVHM) by modifying±1 points [10]:

$$mvx'_{ij} = \begin{cases} mvx_{ij} & if\left(mvx_{ij} = 0\ or\ \mathrm{mod}(i,K) = 0\right) \\ \dfrac{4mvx_{ij} + sign\left(mvx_{ij}\right)}{4} & if\left(|4mvx_{ij}| > 1\ and\ Key(i) = 1\right) \\ \dfrac{4mvx_{ij} + sign\left(mvx_{ij}\right)b}{4} & if\left(|4mvx_{ij}| = 1\ and\ Key(i) = 1\right) \end{cases} \tag{1}$$

$$mvy'_{ij} = \begin{cases} mvy_{ij} & if\left(mvy_{ij} = 0\ or\ \mathrm{mod}(i,K) = 0\right) \\ \dfrac{4mvy_{ij} + sign\left(mvy_{ij}\right)}{4} & if\left(|4mvy_{ij}| > 1\ and\ Key(i) = 0\right) \\ \dfrac{4mvy_{ij} + sign\left(mvy_{ij}\right)b}{4} & if\left(|4mvy_{ij}| = 1\ and\ Key(i) = 0\right) \end{cases} \tag{2}$$

where mvx and mvy are the horizontal and vertical components of the motion vector, respectively, the mvx' and mvy' are the component after modified, respectively. Then mv_{ij} is the j-th motion vector of the i-th frame. K is a parameter that used to control the frame interval, and i is the number of current frame. When mod(i, K) = 0, it means that frame f_i is the reference frame of f_{i+1} to f_{i+k}. b is the to-be embedded data bit. Key is a random binary sequence, and the length is equal to the number of frames that to be embedded, the value of $Key(i)$ is used to determine which component of motion vector is employed for data embedding.

The data extracting algorithm:

$$b' = \begin{cases} 0 & \begin{array}{l} if\left(\left(\left(Key(i) = 1\ and\ |4mvx'| = 1\right)\right.\right. \\ \left.\left. or\ \left(Key(i) = 0\ and\ |4mvy'| = 1\right)\right)\right. \\ \left. and\ (mod(i,K) \neq 0)\right) \end{array} \\ 1 & \begin{array}{l} if\left(\left(\left(Key(i) = 1\ and\ |4mvx'| = 2\right)\right.\right. \\ \left.\left. or\ \left(Key(i) = 0\ and\ |4mvy'| = 2\right)\right)\right. \\ \left. and\ (mod(i,K) \neq 0)\right) \end{array} \end{cases} \tag{3}$$

As the algorithm employs the method proposed by Ni et al. [9], the embedding process is reversible. The video recovery algorithm is as follows:

$$mvx''_{ij} = \begin{cases} mvx'_{ij} & if\left(mvx'_{ij} = 0 \; or \; Key(i) = 0 \right. \\ & \left. or \; mod(i, K) = 0 \; or \; |4mvx'_{ij}| = 1\right) \\ \dfrac{4mvx'_{ij} - sign\left(mvx'_{ij}\right)}{4} & if\left(|4mvx'_{ij}| \geq 2 \right. \\ & \left. and \; Key(i) = 1\right) \end{cases} \tag{4}$$

$$mvy''_{ij} = \begin{cases} mvy'_{ij} & if\left(mvy'_{ij} = 0 \; or \; Key(i) = 1 \right. \\ & \left. or \; mod(i, K) = 0 \; or \; |4mvy'_{ij}| = 1\right) \\ \dfrac{4mvy'_{ij} - sign\left(mvy'_{ij}\right)}{4} & if\left(|4mvy'_{ij}| \geq 2 \right. \\ & \left. and \; Key(i) = 0\right) \end{cases} \tag{5}$$

Obviously, the embedding capacity of the previous algorithm is depended on the number of 1 and -1 points, and the distortion is limited in K frames.

3 Proposed algorithm

From the previous algorithm, we can conclude that only one-bit data can be embedded when the motion vector changed once, which limits the capacity. In order to improve the embedding efficiency, we present a novel algorithm based on two- dimensional histogram modification (2DMVHM) in this paper, shown in Fig. 2. m_t is the t-th data bit to be embedded, m_{t+1} is the $(t+1)$-th data bit to be embedded.

The algorithm can be described as follows:

(1) Encode the video sequence by H.264 standard, in the process of coding, select the motion vectors used for data embedding.

(2) For every selected motion vector, get the horizontal and vertical component, denoted as (mvx, mvy), which makes up an embedding pair, before embedding, all the pairs would be divided into 17 nonintersecting sets, and each set has an embedding algorithm.

(3) According to which set the embedding pair belongs to, the corresponding embedding algorithm is used to modify the value of the pair, and data would be embedded at the same time.

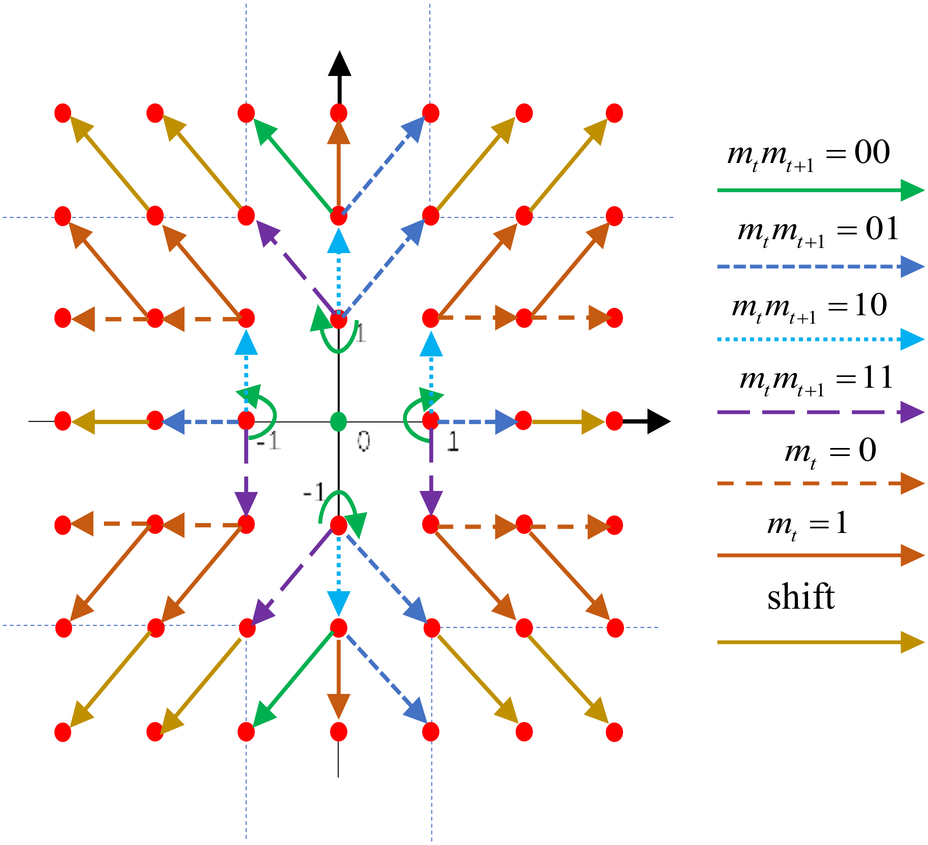


Fig. 2 embedding algorithm of 2DMVHM

3.1 Sets classification

As shown in Fig. 2, all the points in the figure would be divided into 17 nonintersecting sets, the sets are defined as follows:

$$A_1 = \{(1, 0)\}$$
$$A_2 = \{(0, 1)\}$$
$$A_3 = \{(-1, 0)\}$$
$$A_4 = \{(0, -1)\}$$
$$A_5 = \{(Y_{S1}, 0)|Y_{S1} > 1\}$$
$$A_6 = \{(Y_{S1}, 0)|Y_{S1} < -1\}$$
$$A_7 = \{(Y_{S1}, 1)|Y_{S1} \geq 1\}$$
$$A_8 = \{(Y_{S1}, 1)|Y_{S1} \leq -1\}$$
$$A_9 = \{(Y_{S1}, -1)|Y_{S1} \leq -1\}$$
$$A_{10} = \{(Y_{S1}, -1)|Y_{S1} \geq 1\}$$
$$A_{11} = \{(0, Y_{S2})|Y_{S2} > 1\}$$
$$A_{12} = \{(0, Y_{S2})|Y_{S2} < -1\}$$
$$A_{13} = \{(Y_{S1}, Y_{S2})|Y_{S1} \geq 1, Y_{S2} \geq 2\}$$
$$A_{14} = \{(Y_{S1}, Y_{S2})|Y_{S1} \leq -1, Y_{S2} \geq 2\}$$
$$A_{15} = \{(Y_{S1}, Y_{S2})|Y_{S1} \leq -1, Y_{S2} \leq -2\}$$
$$A_{16} = \{(Y_{S1}, Y_{S2})|Y_{S1} \geq 1, Y_{S2} \leq -2\}$$
$$A_{17} = \{(0, 0)\}$$

Among them, (Y_{S1}, Y_{S2}) is four times of *mvx* and *mvy*, respectively. Because the accuracy of motion estimation is 1/4 pixel in H.264, and in the below text, $\left(Y'_{S1}, Y'_{S2}\right)$ is four times of *mvx*′ and *mvy*′, respectively.

3.2 Embedding algorithm

The presented algorithm can be illustrated by Fig. 2. The details of embedding procedure are described as follows:

(1) if $(Y_{S1}, Y_{S2}) \in A_1$, the marked components denoted as $\left(Y'_{S1}, Y'_{S2}\right)$ will be

$$\left(Y'_{S1}, Y'_{S2}\right) = \begin{cases} (Y_{S1}, Y_{S2}) & \textit{if } m_t m_{t+1} = 00 \\ (Y_{S1}+1, Y_{S2}) & \textit{if } m_t m_{t+1} = 01 \\ (Y_{S1}, Y_{S2}+1) & \textit{if } m_t m_{t+1} = 10 \\ (Y_{S1}, Y_{S2}-1) & \textit{if } m_t m_{t+1} = 11 \end{cases} \tag{6}$$

(2) if $(Y_{S1}, Y_{S2}) \in A_2$, then

$$\left(Y'_{S1}, Y'_{S2}\right) = \begin{cases} (Y_{S1}, Y_{S2}) & \textit{if } m_t m_{t+1} = 00 \\ (Y_{S1}+1, Y_{S2}+1) & \textit{if } m_t m_{t+1} = 01 \\ (Y_{S1}, Y_{S2}+1) & \textit{if } m_t m_{t+1} = 10 \\ (Y_{S1}-1, Y_{S2}+1) & \textit{if } m_t m_{t+1} = 11 \end{cases} \tag{7}$$

(3) if $(Y_{S1}, Y_{S2}) \in A_3$, then

$$\left(Y'_{S1}, Y'_{S2}\right) = \begin{cases} (Y_{S1}, Y_{S2}) & \textit{if } m_t m_{t+1} = 00 \\ (Y_{S1}-1, Y_{S2}) & \textit{if } m_t m_{t+1} = 01 \\ (Y_{S1}, Y_{S2}+1) & \textit{if } m_t m_{t+1} = 10 \\ (Y_{S1}, Y_{S2}-1) & \textit{if } m_t m_{t+1} = 11 \end{cases} \tag{8}$$

(4) if $(Y_{S1}, Y_{S2}) \in A_4$, then

$$\left(Y'_{S1}, Y'_{S2}\right) = \begin{cases} (Y_{S1}, Y_{S2}) & \textit{if } m_t m_{t+1} = 00 \\ (Y_{S1}+1, Y_{S2}-1) & \textit{if } m_t m_{t+1} = 01 \\ (Y_{S1}, Y_{S2}-1) & \textit{if } m_t m_{t+1} = 10 \\ (Y_{S1}-1, Y_{S2}-1) & \textit{if } m_t m_{t+1} = 11 \end{cases} \tag{9}$$

(5) if $(Y_{S1}, Y_{S2}) \in A_7$, then

$$\left(Y'_{S1}, Y'_{S2}\right) = \begin{cases} (Y_{S1}+1, Y_{S2}) & \textit{if } m_t = 0 \\ (Y_{S1}+1, Y_{S2}+1) & \textit{if } m_t = 1 \end{cases} \tag{10}$$

(6) if $(Y_{S1}, Y_{S2}) \in A_8$, then

$$\left(Y'_{S1}, Y'_{S2}\right) = \begin{cases} (Y_{S1}-1, Y_{S2}) & \textit{if } m_t = 0 \\ (Y_{S1}-1, Y_{S2}+1) & \textit{if } m_t = 1 \end{cases} \tag{11}$$

(7) if $(Y_{S1}, Y_{S2}) \in A_9$, then

$$\left(Y'_{S1}, Y'_{S2}\right) = \begin{cases} (Y_{S1}-1, Y_{S2}) & if\ m_t = 0 \\ (Y_{S1}-1, Y_{S2}-1) & if\ m_t = 1 \end{cases} \tag{12}$$

(8) if $(Y_{S1}, Y_{S2}) \in A_{10}$, then

$$\left(Y'_{S1}, Y'_{S2}\right) = \begin{cases} (Y_{S1}+1, Y_{S2}) & if\ m_t = 0 \\ (Y_{S1}+1, Y_{S2}-1) & if\ m_t = 1 \end{cases} \tag{13}$$

(9) if $(Y_{S1}, Y_{S2}) \in A_{11}$, then

$$\left(Y'_{S1}, Y'_{S2}\right) = \begin{cases} (Y_{S1}, Y_{S2}+1) & if\ m_t = 1 \\ (Y_{S1}+1, Y_{S2}+1) & if\ m_t m_{t+1} = 01 \\ (Y_{S1}-1, Y_{S2}+1) & if\ m_t m_{t+1} = 00 \end{cases} \tag{14}$$

(10) if $(Y_{S1}, Y_{S2}) \in A_{12}$, then

$$\left(Y'_{S1}, Y'_{S2}\right) = \begin{cases} (Y_{S1}, Y_{S2}-1) & if\ m_t = 1 \\ (Y_{S1}+1, Y_{S2}-1) & if\ m_t m_{t+1} = 01 \\ (Y_{S1}-1, Y_{S2}-1) & if\ m_t m_{t+1} = 00 \end{cases} \tag{15}$$

(11) if (Y_{S1}, Y_{S2}) belongs to A_5, A_6, A_{13}, A_{14}, A_{15}, A_{16}, no information is embedded into the MVs, and $\left(Y'_{S1}, Y'_{S2}\right)$ will be

$$\left(Y'_{S1}, Y'_{S2}\right) = \begin{cases} (Y_{S1}+1, Y_{S2}) & if\ (Y_{S1}, Y_{S2}) \in A_5 \\ (Y_{S1}-1, Y_{S2}) & if\ (Y_{S1}, Y_{S2}) \in A_6 \\ (Y_{S1}+1, Y_{S2}+1) & if\ (Y_{S1}, Y_{S2}) \in A_{13} \\ (Y_{S1}-1, Y_{S2}+1) & if\ (Y_{S1}, Y_{S2}) \in A_{14} \\ (Y_{S1}-1, Y_{S2}-1) & if\ (Y_{S1}, Y_{S2}) \in A_{15} \\ (Y_{S1}+1, Y_{S2}-1) & if\ (Y_{S1}, Y_{S2}) \in A_{16} \end{cases} \tag{16}$$

(12) if $(Y_{S1}, Y_{S2}) \in A_{17}$, then $\left(Y'_{S1}, Y'_{S2}\right)$ will be

$$\left(Y'_{S1}, Y'_{S2}\right) = (Y_{S1}, Y_{S2}) \quad if\ (Y_{S1}, Y_{S2}) \in A_{17} \tag{17}$$

After modified, the values of the pair will belong to a new set, whereas, from Fig. 2, we can see that the in-degree of each point (except for (0,0)) is 1, and it means after modified, every marked point comes from the unique point. That is to say, the algorithm is reversible, the extracting algorithm is the reverse process of Fig. 2.

4 Experimental results and analysis

4.1 Data embedding

The original video can be generated by encoding YUV video with H.264 standard, and we take 8 video sequences for experiments. We will take 150 frames from each video for data embedding.

In the experiment, we will compare the previous algorithm MVHM with presented algorithm 2DMVHM in the aspects of embedding capacity and PSNR. The comprehensive experimental results are shown in Table 1.

From Table 1, we can see that different video sequence provides different embedding capacity, and the capacity will rise with the increase of K, whereas, the PSNR is decreasing as K increases, that's because when the frame interval is larger, more frames will be embedded information, and the distortion will accumulate more severely.

Compared with the previous algorithm, the method proposed in this paper possesses a higher embedding capacity, and the distortion performance is acceptable. The reasons are as follows:(1) for the previous algorithm, under the control of *Key*(*i*), only one kind of components will be selected to embed data, but in the new method, *mvx* and *mvy* are modified together, and when the pair is changed once, 2 bits can be embedded at most, and due to the introduction of two-dimension, some original points that are unable to embed information (in the previous method) can be employed to hide data (in the new method), like (1,0). (2) for the distortion performance of 2DMVHM method, although the two components of MVs are modified together, the results turn out to be only one of them would be changed at most time. Take (1, 0) for instance, after modified, the result would be one of the four, (1, 0), (2, 0), (1, 1), (1, −1), and only one component would be changed at a time.

In consideration of sometimes PSNR cannot reflect the distortion completely, we conducted additional experiments to figure out the bitrate increase introduced by the proposed algorithm. As shown in Table 2, under different K, the bitrate increase differs from each other. We can see that under the same K, 2DMVHM algorithm suffers a larger bitrate increase compared with MVHM. After the value of MV being modified, the residual between the reference block and current block tends to become larger, it will cost more bits to encode the residual. On the other hand, in the experiment, the first frame is I-mode frame, and the subsequent frames are all P-mode, if more I-mode frames are encoded, the bitrate variation will be decreased, but the embedding capacity will be decreased as well.

Table 1 comprehensive experimental results

Video		2DMVHM		MVHM		Original video
		PSNR(dB)	capacity(bit)	PSNR(dB)	capacity(bit)	PSNR(dB)
News	K = 2	37.66	3634	38.28	1159	39.15
	K = 3	37.21	5666	38.19	1846	
	K = 5	36.39	7960	37.79	2932	
	K = 7	36.12	8788	37.66	2975	
Mobile	K = 2	32.07	11,874	33.98	5780	36.58
	K = 3	30.38	12,151	32.71	6354	
	K = 5	29.00	10,229	31.59	5694	
	K = 7	28.46	8875	31.15	4980	
City	K = 2	36.52	3495	37.43	1780	38.02
	K = 3	35.49	4161	36.78	2508	
	K = 5	34.61	3582	36.05	2028	
	K = 7	34.29	3409	35.71	2175	
Carphone	K = 2	37.57	8252	38.66	3335	39.47
	K = 3	36.62	10,504	38.05	4266	
	K = 5	35.93	11,659	37.58	4650	
	K = 7	35.63	11,682	37.34	4780	
Salesman	K = 2	37.41	4338	37.76	1699	37.79
	K = 3	37.04	6244	37.58	2320	
	K = 5	36.61	8008	37.35	3134	
	K = 7	36.34	9304	37.16	3525	
Coastguard	K = 2	34.28	11,351	35.99	2872	37.00
	K = 3	33.06	12,420	35.34	3732	
	K = 5	31.89	10,857	34.54	5191	
	K = 7	31.47	10,533	34.39	4548	
Container	K = 2	37.18	6626	38.26	2108	38.46
	K = 3	36.36	10,182	38.07	3288	
	K = 5	35.51	12,634	37.68	3996	
	K = 7	35.25	13,250	37.48	3887	
Foreman	K = 2	36.49	8637	37.66	3885	38.57
	K = 3	35.48	10,036	36.93	4645	
	K = 5	34.68	10,240	36.29	4659	
	K = 7	34.19	9100	35.99	4079	

Table 2 comparison of bitrate increase

Video	Bitrate increase(%)					
	2DMVHM			MVHM		
	K = 2	K = 3	K = 5	K = 2	K = 3	K = 5
News	11.16	19.44	33.81	10.84	18.9	33.04
Mobile	5.53	9.26	14.04	5.2	8.81	13.6
City	7.1	24.82	60.61	6.56	24.19	60.08
Carphone	10.33	17.3	29.19	9.63	16.33	28.03
Salesman	8.05	17.37	19.64	7.79	16.92	28.96
Coastguard	17.61	30.82	49.03	17.09	30.14	48.32
Container	6.52	16.45	31.08	6.01	15.64	29.97
Foreman	15.24	27.29	46.31	14.5	26.37	45.28

4.2 Data extracting and video recovery

By decoding the data embedded video, receiver can get all the motion vectors and different marked motion vectors have different extracting algorithms, data can be extracted by using the corresponding algorithm, at the same time, the video can be recovered completely. The data extracting flow chart is shown in Fig. 3.

As the data extracting is the reverse process of algorithm shown in Fig. 2, here, we'll take A_1 for instance:

$$data = \begin{cases} (0,0) & if \left(Y'_{S1} = 1 \; and \; Y'_{S2} = 0\right) \\ (0,1) & if \left(Y'_{S1} = 2 \; and \; Y'_{S2} = 0\right) \\ (1,0) & if \left(Y'_{S1} = 1 \; and \; Y'_{S2} = 1\right) \\ (1,1) & if \left(Y'_{S1} = 1 \; and \; Y'_{S2} = -1\right) \end{cases} \tag{18}$$

In this equation, *data* is the extracted information.

The video will be recovered when data is extracted, we'll take A_{11} for instance:

$$\left(Y''_{S1}, Y''_{S2}\right) = \begin{cases} \left(Y'_{S1}, Y'_{S2}-1\right) & if \left(Y'_{S1} = 0 \; and \; Y'_{S2} > 2\right) \\ \left(Y'_{S1}-1, Y'_{S2}-1\right) & if \left(Y'_{S1} = 1 \; and \; Y'_{S2} > 2\right) \\ \left(Y'_{S1}+1, Y'_{S2}-1\right) & if \left(Y'_{S1} = -1 \; and \; Y'_{S2} > 2\right) \end{cases} \tag{19}$$

In the equation, $\left(Y''_{S1}, Y''_{S2}\right)$ is four times of recovered *mvx* and *mvy*, respectively.

By this way, the data can be extracted while the video will be recovered. For the extracting experiment, we embed data into 150 frames of the video "foreman.qcif", and let K is 2, then decode the sequence with different Ks, in the experiments, we would take the original video as comparison. The result is shown in Fig. 4. From the figure we can conclude that the data-embedded video could be recovered only when K is 2, with incorrect K, the quality of video is declining seriously, that's because data embedding influences the quality of video greatly [16, 17].

In the experiments, the video is fully embedded, but in the practical application, there won't be so much information to be embedded, in order to improve the quality of data-embedded

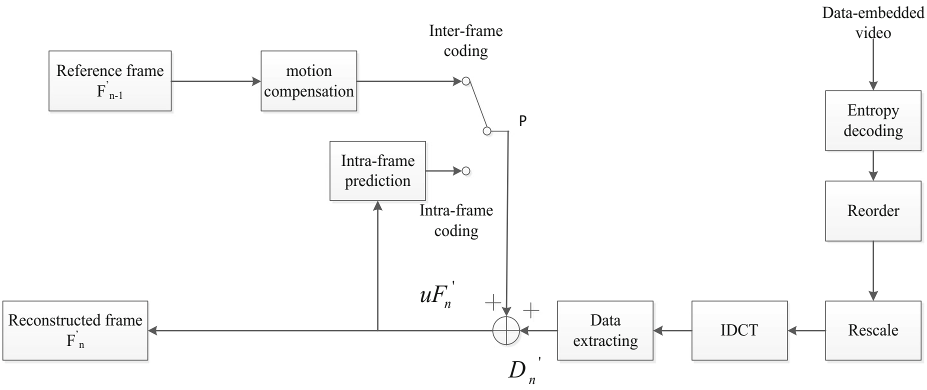


Fig. 3 Data extracting and video recovered process

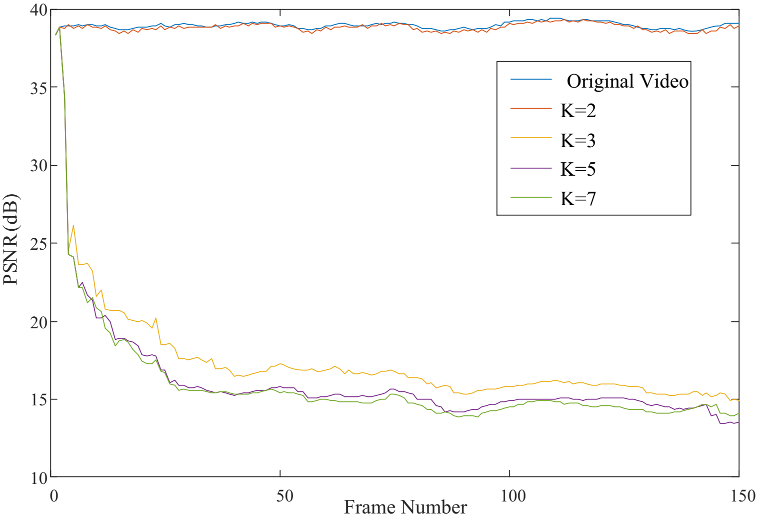


Fig. 4 The results of video recovering

video, we can introduce a parameter Key, which is a random binary sequence used to decide whether the data would be embedded into the current frame or not, that is to say, if Key(i) = 1, the current frame would be used to embed data, otherwise this frame will be kept intact. By this way, the distortion will be decreased ulteriorly.

4.3 Analysis of reversibility

From the data extracting experiments, we can see that the video can be recovered completely after data being extracted. In most instances, the reversibility of an algorithm cannot be ensured until the data extracting experiments have been conducted. By studying the diagram of embedding algorithm shown in Fig. 2, from the perspective of in-degree and out-degree, we presented some rules that could be used to judge the reversibility of an algorithm directly:

(1) In-degree of all the points is 1.
(2) Out-degree of all the points is N.
(3) Where there is an in-degree, there is an out-degree.

When the diagram of an algorithm satisfies the above rules, the algorithm must be reversible. We will take Fig. 1 for instance. Except for the point (0,0), the in-degree of the rest points is 1, it means after modified, we can still find out the original position that the points belong to. The number of out-degree directly influences the embedding capacity of a point. After modified, the points would be changed to a new position, to ensure the reversibility, the modified points must be set apart from the original points that in the same position. It is obviously that the proposed algorithm satisfies all the above rules, the algorithm is reversible in theory, from Fig. 4, we can see that the video can be recovered completely by using the correct parameter K, the PSNR of recovered video and the original video are coincident, it means the proposed method is reversible not only in theory, but also in experiment. There are still many characters can be inferred from the diagram, the relevant research will be implemented in our future work.

5 Conclusion

In this paper, we proposed a novel two-dimensional histogram modification based RDH for H.264, compared with the previous algorithm, the proposed method can make full use of relevance between the two components of MV, and embedding capacity is enlarged greatly, at the same time, the distortion performance is acceptable. However, there is still a large space to improve the performance of the proposed algorithm, and the method could also be extended to multi-dimension in theory.

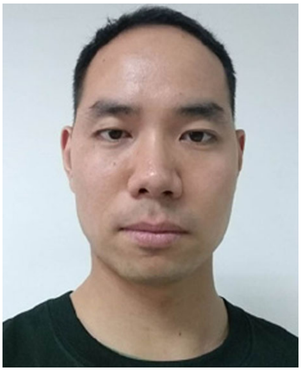
As the cloud computing technology developing greatly, the security problem of cloud storage has drawn the public attention [5], it is significant to apply RDH algorithms into encryption domain to meet the demands of cloud storage. With the developing of steganalysis [8, 14, 20], the conventional data hiding algorithms are faced with severe challenges, coverless information hiding will provide a promising solution [4]. In our future work, we would like to study the common characters that RDH algorithms share, and try to build a theoretical base for the design of RDH methods, then we would try to design a RDH method suitable for encrypted video, and the coverless information hiding based on video will also be studied.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

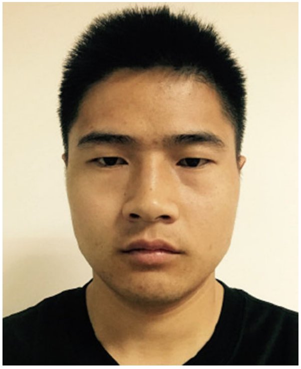
1. Alattar AM (2004) Reversible watermark using the difference expansion of a generalized integer transform. IEEE Trans Image Process 13(8):1147–1156
2. Al-Qershi OM, Khoo BE (2013) Two-dimensional difference encryption (2d-de) scheme with a characteristics-based threshold. Signal Process 93(1):154–162
3. Barton JM (1997) Method and apparatus for embedding authentication information within digital data. US Patent US5646997
4. Cao Y, Zhou ZL, Sun XM, Gao CZ (2018) Coverless information hiding based on the molecular structure images of material. CMC 54(2):197–207
5. Hwang K, Li D (2010) Trusted cloud computing with secure resources and data coloring. IEEE Internet Comput 14(5):14–22
6. Lin KZ, Wei Y, Bo F (2008) Digital watermarking synchronization algorithm of MPEG-4 video. Int Multi-Sympo Comput Comput Sci:164–167
7. Liu YX, Ju LM, Hu MS, Ma XJ, Zhao HG (2015) A robust reversible data hiding scheme for H.264 without distortion drift. Neurocomputing 151:1053–1062
8. Ma YY, Luo XY, Li XL, Bao ZK, Zhang Y (2018) Selection of rich model Steganalysis features based on decision rough set α-positive region reduction. IEEE Trans Circ Syst Video Technol. https://doi.org/10.1109/TCSVT.2018.2799243
9. Ni ZC, Shi YQ, Ansari N, Su WF (2006) Reversible data hiding. IEEE Trans Circ Syst Video Technol 16(3):354–362
10. Niu K, Yang XY, Zhang YN (2017) A novel video reversible data hiding algorithm using motion vector for H.264/AVC. Tsinghua Sci Technol 22(5):489–498
11. Richardson IEG (2003) H.264 and MPEG-4 video compression: video coding for next-generation multimedia. Wiley, UK, p 171
12. Song GH, Li ZT, Zhao J, Hu J, Tu H (2015) A reversible video steganography algorithm for MVC based on motion vector. Multimed Tools Appl 74:3759–3782
13. Tian J (2003) Reversible data embedding using a difference expansion. IEEE Trans Circuits Syst Video Technol 13(8):890–896
14. Wang JW, Li T, Shi YQ, Lian SG, Ye JY (2017) Forensics feature analysis in quaternion wavelet domain for distinguishing photographic images and computer graphics. Multim Tools Appl 76(22):23721–23737
15. Xu DW, Wang RD, Shi YQ (2014) An improved reversible data hiding-based approach for intra-frame error concealment in H.264/AVC. J Vis Commun Image R 25:410–422

16. Zeng XA, Chen ZY, Chen M, Xong Z (2011) Reversible video watermarking using motion estimation and prediction error expansion. J Inf Sci Eng 27(2):465–479
17. Zeng X, Chen Z, Xiong Z (2011) Issues and solution on distortion drift in reversible video data hiding. Multimedia Tools Appl 52(2–3):465–484
18. Zhang Y, Qin C, Zhang WM, Liu FL, Luo XY (2018) On the fault-tolerant performance for a class of robust image steganography. Signal Process 146:99–111
19. Zhao J, Li ZT, Feng B (2016) A novel two-dimensional histogram modification for reversible data embedding into stereo H.264 video[J]. Multimed Tools Appl 75:5959–5980. https://doi.org/10.1007/s11042-015-2558-9
20. Zhou QL, Qiu YB, Lu JF, Yuan WQ, Feng XQ, Mao XY (2018) Steganography using reversible texture synthesis based on seeded region growing and LSB. CMC 55(1):151–163



Dong Li received his Bachelor degree from Engineering University of People's Armed Police in 2016. He is currently a M.S. candidate in Key Laboratory of Network & Information Security of People's Armed Police, Department of Electronic Technology in Engineering University of People's Armed Police. His research interests involve in Information Hiding and Video Processing.

YingNan Zhang received his M.S. degree from Engineering University of People's Armed Police in 2014. He is currently a Ph.D. candidate in Key Laboratory of Network & Information Security of People's Armed Police, Department of Electronic Technology in Engineering University of People's Armed Police. His research interests involve in Information Hiding and Video Processing.



XinChao Li received his Bachelor degree from Engineering University of People's Armed Police in 2016. He is currently a M.S. candidate in Key Laboratory of Network & Information Security of People's Armed Police, Department of Electronic Technology in Engineering University of People's Armed Police. His research interests involve in Cryptography and Information Security.

Ke Niu received his M.S. degree from Engineering University of People's Armed Police in 2007. He is currently an associate professor, and a Ph.D. candidate in Key Laboratory of Network & Information Security of People's Armed Police, Electronic Technology Department in Engineering University of People's Armed Police. His research interests involve in Information Hiding and Video Processing.

XiaoYuan Yang received M.S. degree from Xidian University in 1991. He has been a Professor, Ph.D. advisor in Engineering University of People's Armed Police. His research interests include Cryptography and Information Security.

YuJuan Sun received her Ph.D. degree in mathematics from Lanzhou University, Lanzhou, China, in 2010. She is currently with Xidian University, Xi'an, China. Her research interests include cryptography and coding theory.