Multimed Tools Appl (2015) 74:3759–3782
DOI 10.1007/s11042-013-1798-9

A reversible video steganography algorithm for MVC based on motion vector

Guanghua Song · Zhitang Li · Juan Zhao · Jun Hu · Hao Tu

Published online: 14 December 2013

Abstract In this paper we present a reversible video steganography scheme for hiding secret data into the motion vector of each block in 3D MVC videos. Under this approach the idea of the inner product is introduced to achieve reversibility. By establishing the inner product between the motion vector and the modulation vector and setting the embedding conditions, we embed 1 bit data into each motion vector and the proposed algorithm is reversible. Moreover, in order to avoid distortion drift, we only embed data into b4-frames with the coding feature of 3D MVC videos. Experimental results also confirm that the proposed scheme can provide expected acceptable video quality of stegovideos and successfully achieve reversibility.

Keywords Reversible video steganography · Multi-view coding · Motion vector · Inner product · Distortion drift

1 Introduction

Data hiding is referred to as a process to hide data into the cover media [4]. In most cases, the cover media will be affected by some distortion after data hiding and the processed media cannot be converted back to the original one. However, in some applications, such as medical diagnosis, military images, remote sensing image processing, legal certification and evidence and other fields, it is critical to restore the marked media back to the original media [6]. And the reversible video steganography techniques are used in a variety of domains at present.

Recently, some reversible steganography techniques have been reported in some literatures. The first reversible data hiding algorithm was the patent submitted by Bart in 1994 [3]. After

G. Song · Z. Li (✉) · J. Zhao · J. Hu · H. Tu
Department of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China
e-mail: leeying@mail.hust.edu.cn

G. Song
e-mail: ghsong@hust.edu.cn

Z. Li · H. Tu
Network and Computing Center, Huazhong University of Science and Technology, Wuhan 430074, China

that, the algorithms about reversible data hiding were constantly emerging. In 2003, Tian [30] proposed a difference-expansion (DE) method without compressing the original medium. The algorithm divided the image into pairs of pixels and classified the pairs into three groups: changeable, expandable and exceptional. The method achieved reversibility by the correlation of adjacent pixels. The DE algorithm was researched by many other researchers [1, 7, 13–15, 29], and had a profound impact on reversible steganography technology development. After the DE algorithm, Ni etc. [24] presented another representative reversible information hiding algorithm called reversible algorithm based on the histogram shifting (HS). This algorithm utilized the zero or the minimum points of the histogram of an image and slightly modified the pixel brightness levels to embed data into the image. In the histogram-based data hiding algorithm, the number of pixels in the peak point was the maximal hiding capacity for the secret message to be embedded. More of the peak and zero pairs were selected to enhance the hiding capacity. Afterwards, researchers proposed many improved algorithms based on the original HS algorithm [8, 16, 28, 31].

Currently the research on reversible steganography algorithm for H.264/AVC mainly focused on 2D fields, and most of the existing robust algorithms embedded the data into the DCT coefficients of I-frames [19, 25, 27, 34]. In [27], the authors presented a new reversible data hiding algorithm based on integer transform and adaptive embedding. It can embed as high as 2.17 bits per pixel into Lena image with a reasonable PSNR of 20.71 dB. In [19, 25, 34], the authors discussed a variety of robust data hiding algorithms in detail. There were also many other methods that use other coefficients such as DWT, VQ, motion vector (MV) and so on [5, 9, 10, 22, 26, 33]. In [22], a prescription-based error concealment (PEC) method was studied, and the proposed method was capable of achieving PSNR improvement of up to 1.48 dB, at a considerable bit-rate, when the packet loss rate was 20 %. In [9], the authors researched a novel high capacity reversible image data hiding scheme using a prediction technique which was effective for error resilience in H.264/AVC. The proposed method, called shifted intra prediction error (SIPE), was able to hide more secret data while the PSNR of the marked image was about 48 dB. Many other methods were presented in [5, 10, 26, 33]. In previous work, most of the algorithms inevitably encountered distortion drift problem including intra and inter. And all the mentioned algorithms were only for 2D fields.

In recent years, with the advancement of 3D video technologies and rapid communication network deployment, 3D videos such as 3D films are becoming more and more popular. Therefore, traditional steganography algorithm researchers began to turn their attention to the new cover media. In 2010, Joint Video Team (JVT) released the Multi-View Coding (MVC) [12] standard based on H.264/AVC, and wrote them as an appendix in the form of the latest H.264/AVC. JVT adopted the structure of hierarchical B-frames proposed in [23] which included the intra prediction, the inter prediction and the inter-view prediction. The hierarchical B-frames prediction structure with two viewpoints was as shown in Fig. 1. Compared to the 2D videos, the

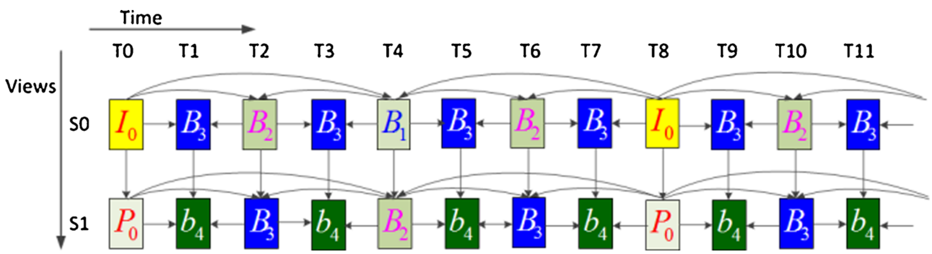

Fig. 1 Hierarchical B-frames prediction structure with two viewpoints [23]

3D videos are generated by the two-way (or more) 2D videos named $S0$ and $S1$, and the prediction direction is from $S0$ to $S1$. So, we can only embed data into the $S1$ viewpoint to prevent drift between the viewpoints. This will achieve better visual effects and imperceptibility.

As shown in Fig. 1, it is clear that there is only one I-frame and one P-frame in each GOP with the coding structure of 3D MVC videos, the remaining are all B-frames [23]. Further, as I-frames and P-frames are all key references, modifying the I-frames or P-frames will inevitably lead to inter drift distortion. Therefore, we take the B-frames as the candidate for data embedding, and the most abundant information in the B-frames locates in motion vectors (MV). The MV is a very attractive choice to embed secret data, as it is relatively easy to manipulate compared with residuals or other syntax elements. From some previous work, we found that altering MVs did not necessarily lead to significant videos quality degradation, if the MV was carefully chosen by [2, 11, 20]. In addition to the standard frame structure, as shown in Fig. 1, there are several other coding configurations used in the encoder to compress a video. We will elaborate this in Section 2.2.

The idea of using motion vectors (MVs) as the covert data carrier can be dated back to Kutter et al's work [21] in which they proposed a video watermark scheme by altering the MVs directly. In recent years, some researchers were involved in this area, and achieved a few results included in [18, 35, 36]. However, these algorithms are not reversible. In this paper, by combining MVC coding features, we propose a new reversible video steganography algorithm based on motion vectors without inter distortion drift. In order to prevent the inter distortion drift problem, we hide data into the b4-frames. More importantly, by modifying the motion vectors, the idea of the inner product is used to achieve reversibility. Given a motion vector, we can embed 1 bit data into each motion vector. The complexity of the algorithm is lower and the embedding capacity is greater than the current algorithms. To the best of our knowledge, the proposed algorithm is new in that:

- Unlike most of the existing algorithms, we hide data into the motion vector of b4-frames. This is more suitable for the MVC videos with the structure of hierarchical B-frames.
- Establishing the inner product between the motion vector and the modulation vector to achieve the reversible steganography is a new attempt.
- The proposed algorithm takes MVC videos as the carrier. As far as we know, this is the first reversible steganography algorithm for MVC videos.

The remainder of this paper is organized as below. First, we briefly review some previous work in Section 2. Then, in Section 3, we present our new reversible embedding scheme for MVC videos. Experimental results are listed in Section 4. Finally, the conclusions are presented in Section 5.

2 Motion vector and the prediction structure of MVC videos

2.1 Motion compensation and motion vector

Motion compensation exploits the fact that, often, for many frames of a movie, the only difference between one frame and the subsequent one is the result of the movement of the camera or an object in the frame. Therefore, much of the information that represents one frame is the same as the information of the next frame. Using motion compensation, a video stream contains some full (reference) frames and the information that used to transform the previous frame into the next frame. In block motion compensation, the frames are partitioned into blocks of pixels (e.g.

macroblocks of 4×4 pixels in H.264). Each block is predicted from a block with the same size in the reference frame. The blocks are not transformed into any way apart from being shifted to the position of the predicted block. This shift is represented by a motion vector [17].

A generic structure of inter-MB coding is depicted in Fig. 2. Once B_r is chosen, B's MV will be calculated as:

$$MV = \left(mv_x, mv_y\right) = (x_r - x, y_r - y) \tag{1}$$

Where mv_x, mv_y are the horizontal and vertical components respectively, (x_r, y_r) and (x, y) denote the coordinates of B_r and B respectively.

Motion vector based watermarking was first introduced by Kutter et al. [21]. The idea was to change motion vectors so that a binary password could be embedded into the motion vectors. To be specific, they modified the LSBs of motion vectors to embed a watermark so that the parity of the motion vectors satisfied the watermark bit value. However, the algorithm was not reversible. After extracting the hidden data, we cannot recover the original videos. Also most of the existing algorithms only used one component of the motion vector and ignored the vector attribute of the motion vectors.

Later in Section 3, an optimized method based on motion vectors is introduced, and the method is reversible.

2.2 The prediction structure of 3D videos

JVT adopted the structure of hierarchical B-frames proposed in [23] which combined intra prediction, inter prediction and inter-view prediction. There are only one I-frame and one P-frame in a GOP in 3D videos with two viewpoints, as shown in Fig. 1. As the key frame, I-frames will be referenced by all the remaining frames. If we modify the I-frames, the distortion will be passed to all the rest of the frames, which will cause serious inter-frames distortion drift. Therefore, we generally do not take the I-frames as the carrier. In addition to the I-frames, there is one P-frame in a GOP, and the remaining are B-frames.

Through the study of the prediction structure of MVC videos, it is evident that the prediction direction is from $S0$ to $S1$, as shown in Fig. 3. So, we can embed data into the $S1$ viewpoint to prevent drift between the viewpoints, which is proved by the following **Conclusion 1**.

C1. *The inter-view-prediction direction of 3D videos with two viewpoints is from* ***S0*** *to* ***S1****. So if we modify the frames in* ***S0****, the distortion will be passed to not only the frames in* ***S0*** *but also in* ***S1*** *which will cause serious distortion drift. However, if we modify the frame in* ***S1,*** *the*

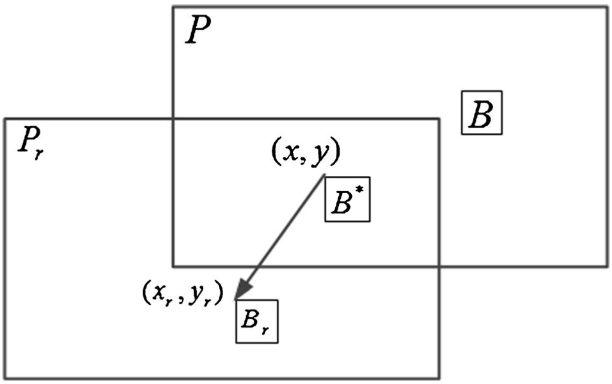

Fig. 2 Motion compensation and motion vector

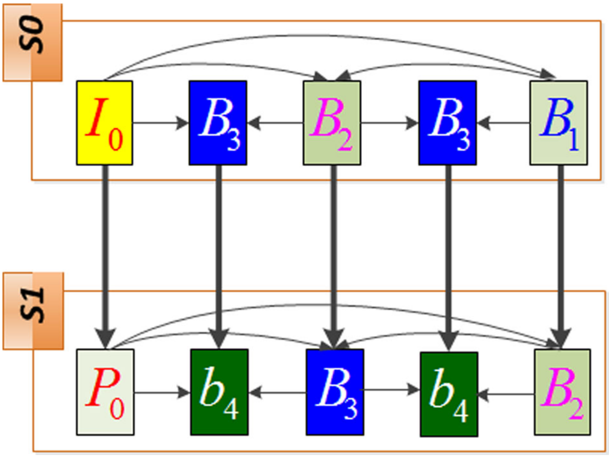

Fig. 3 The prediction direction of 3D videos with two viewpoints

distortion will be passed only to the frames in ***S1****, and the inter-view distortion drift is prevented.*

And we also find that the prediction mode is subject to the rules described in the following **Conclusion 2**.

C2. *There is a non-reference frame in the MVC videos with the structure of hierarchical B-frames in which hiding data will not cause the inter-frame distortion drift.*

As shown in Fig. 1, by examining the b4-frames (e.g. the first b4-frame in a GOP as shown in Fig. 4), we reach the following two conclusions. Firstly, as the frame is located in viewpoint **S1**, it will not be used as a reference frame for inter-view prediction according to **C1**. Secondly, the b4-frame also will not be used as a reference frame for the inter prediction in viewpoint **S1**. So the b4-frames are the non-reference frame in the MVC videos with the structure of hierarchical B-frames. In other words, the b4-frames are neither used as the reference frame for inter prediction nor used as the reference for inter-view prediction.

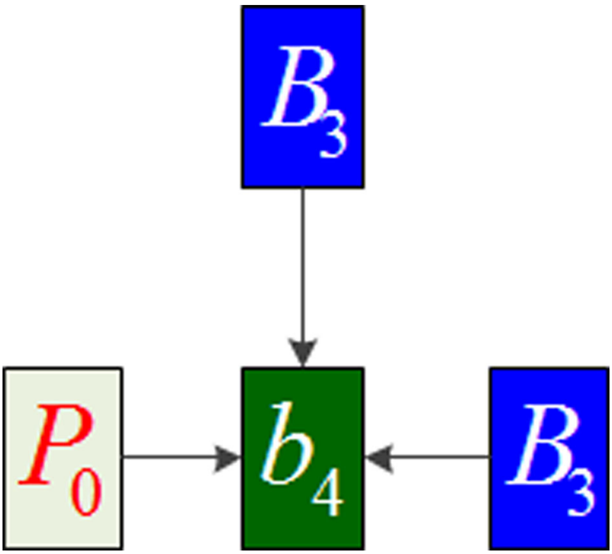

Fig. 4 The first b4-frame in a GOP

Therefore, it will not result in any inter-frame distortion drift when we hide data into b4-frames.

In addition to the standard frame structure shown in Fig. 1, there are several coding configurations that can be used in the encoder to compress a MVC video. For example, in HEVC-based 3D encoder, one can choose one of several possible coding configurations. Each of these configurations can have a different GOP style. However, no matter what kind of frame structure is, the prediction structure of the video is similar. And the prediction direction of 3D videos with two viewpoints is from **S0** to **S1**, so we can hide data into the motion vector of P-frames or B-frames in viewpoint **S1**.

3 The proposed algorithm

In this section, according to the analysis of the coding structure of MVC videos, we propose a data hiding method without inter distortion drift. Then, we introduce the idea of the modulation vector. And we will demonstrate that by establishing the inner product between the motion vector and the modulation vector, we can embed 1 bit data into each motion vector, and the embedding process is reversible.

3.1 The proposed reversible steganography algorithm

Firstly, in order to prevent the problem of inter distortion drift, we hide data into the b4-frames. And we use the idea of the inner product to achieve reversibility. For a given motion vector, we can embed 1 bit data into each motion vector by modifying it. Secondly, we introduce the modulation vector to represent the modification to the motion vector. Thirdly, the proposed reversible video steganography algorithm is given.

Generally, the modulus of *MV* represent the movement intensity, and modifying a large motion vector would not make a great impact on the video quality. In our method, the candidate MB is selected by a predefined threshold T. The modification of the big motion vector would be less perceivable than that of the small motion vector. Another important reason to select motion vectors with big magnitude is that we might have higher probability to find a better motion vector to replace the original motion vector [20].

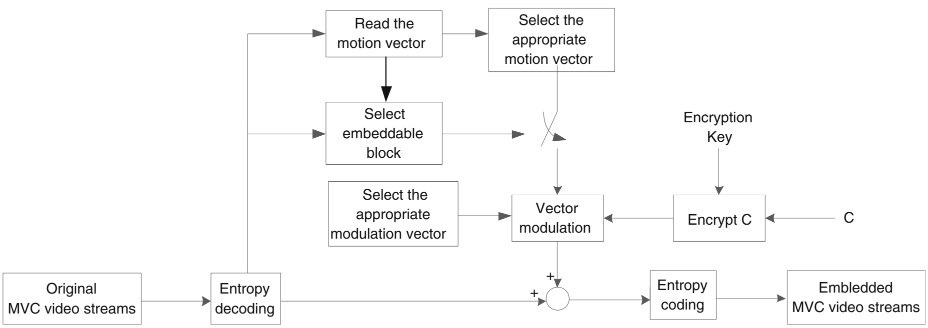

Fig. 5 Proposed data embedding scheme diagram

Therefore, in order to reduce the impact on the video quality, the motion vector is selected according to its modulus by criteria such as $\|MV\|>Threshold(Threshold=0,1,2,\cdots)$.

On the other hand, if we make a large change to motion vector, it will lead to serious decline in the video quality. So we introduce the concept of the modulation vector ($\mathbf{\Delta}$) which indicates the modification to the motion vector.

$$\mathbf{\Delta} = (\Delta \mathrm{x}, \Delta \mathrm{y}) \tag{2}$$

Where $\Delta\mathrm{x}, \Delta\mathrm{y}$ are the horizontal and vertical components of $\mathbf{\Delta}$ respectively.

The modulation vector means the size of modifications of the motion vector. The value of the modulation vector should not be too large in order to avoid the serious decline of the video quality. Therefore, we also need to choose an appropriate modulation vector. And the modulation vector can be as a key to improve the security of the algorithm.

Algorithm 1 Data Hiding in GOP with inner product

Input: Message bitstream C, Original GOP, Threshold, Modulation vector $\mathbf{\Delta}=(\Delta x,\Delta y)$,

Output: Data embedded into the Decoded GOP

encrypt: $C \rightarrow C'$

foreach decoded frame in the GOP do

  if the frame is b4-frames

  then Obtain the motion vector $\mathbf{M}_i=(x,y)$

  foreach Motion vector do

    if $\|\mathbf{M}_i\| > Threshold$

      then

      if $\|\mathbf{M}_i \bullet \mathbf{\Delta}\| < \|\mathbf{\Delta}\|^2$

$$\mathbf{M}_i' = \begin{cases} \mathbf{M}_i + \mathbf{\Delta}, if (C_i' = 1 \,\&\&\, \mathbf{M}_i \bullet \mathbf{\Delta} > 0) \\ \mathbf{M}_i - \mathbf{\Delta}, if (C_i' = 1 \,\&\&\, \mathbf{M}_i \bullet \mathbf{\Delta} \le 0) \\ \mathbf{M}_i, else \end{cases}$$

      else $\|\mathbf{M}_i \bullet \mathbf{\Delta}\| \ge \|\mathbf{\Delta}\|^2$

      not embedding data

$$\mathbf{M}_i' = \begin{cases} \mathbf{M}_i + \mathbf{\Delta}, if (\mathbf{M}_i \bullet \mathbf{\Delta} \ge \|\mathbf{\Delta}\|) \\ \mathbf{M}_i - \mathbf{\Delta}, if (\mathbf{M}_i \bullet \mathbf{\Delta} \le -\|\mathbf{\Delta}\|) \end{cases}$$

until C is completely embedded

end

According to the geometric meaning of the vector inner product, we take $\|\mathbf{M}_i \bullet \mathbf{\Delta}\| < \|\mathbf{\Delta}\|^2$ as a criterion to decide whether to embed or not. This condition would ensure the reversibility of the proposed algorithm. Moreover, in order to improve the security of the algorithm, the embedded data C is encrypted into a binary sequence C'. We assume that $C'=c_1c_2c_3\cdots, c_i\in\{0,1\}$. To avoid the inter distortion drift problem, we embed data into b4-frames (we also hide data into P_0-frames with other frame structure) according to **C1.** Finally, we use the nature of the vector inner products to modulate the motion vectors to achieve reversible steganography. The proposed steganography scheme is shown in **Algorithm 1**.

Figure 5 depicts the scheme of our proposed embedding algorithm. First, the original MVC video is entropy decoded to get the motion vector. Then, the appropriate motion vectors are selected according to the *Threshold*. And we also select an appropriate modulation vectors. The encrypted message is embedded into the appropriate motion vector based on modulo modulation. Then, all the motion vectors are entropy coded to get the target embedded video.

The data extractor extracts the hidden message as a special decoder and our proposal is straightforward, as shown in **Algorithm 2**. After data extraction from the consecutive GOPs, the hidden message is reconstructed back by concatenating the extracted bitstream.

Algorithm 2 Data Extraction

Input: Hidden GOP, Threshold, Modulation vector $\mathbf{\Delta}$
Output: Message bitstream
foreach decoded frame in the GOP do
  if the frame is b4-frames
  then Obtain the motion vector $\mathbf{M}'_i = (x, y)$
   if $\|\mathbf{M}'_i\| > Threshold$
   then
    if $\|\mathbf{M}'_i \bullet \mathbf{\Delta}\| < 2\|\mathbf{\Delta}\|^2$
     if $(\mathbf{M}'_i \bullet \mathbf{\Delta} > \|\mathbf{\Delta}\|^2)$
      $\mathbf{M}_i = \mathbf{M}_i' - \mathbf{\Delta}$
      $C_i' = 1$
     if $(\mathbf{M}'_i \bullet \mathbf{\Delta} \leq -\|\mathbf{\Delta}\|^2)$
      $\mathbf{M}_i = \mathbf{M}_i' + \mathbf{\Delta}$
      $C_i' = 1$
     else
      $\mathbf{M}_i = \mathbf{M}_i'$
      $C_i' = 0$
    else
     not extracted
     $$\mathbf{M}_i = \begin{cases} \mathbf{M}'_i - \mathbf{\Delta}, if\,(\mathbf{M}'_i \bullet \mathbf{\Delta} \geq 2\|\mathbf{\Delta}\|^2) \\ \mathbf{M}'_i + \mathbf{\Delta}, if\,(\mathbf{M}'_i \bullet \mathbf{\Delta} \leq -2\|\mathbf{\Delta}\|^2) \end{cases}$$
 until C' is completely extracted
 decrypt $C' \rightarrow C$
 end

Figure 6 depicts the scheme of our proposed extraction algorithm. First, the embedded MVC video is entropy decoded to get the motion vector. Then, the appropriate motion vectors are selected according to the *Threshold*. And we also select an appropriate modulation vectors. The encrypted messages are extracted from the appropriate

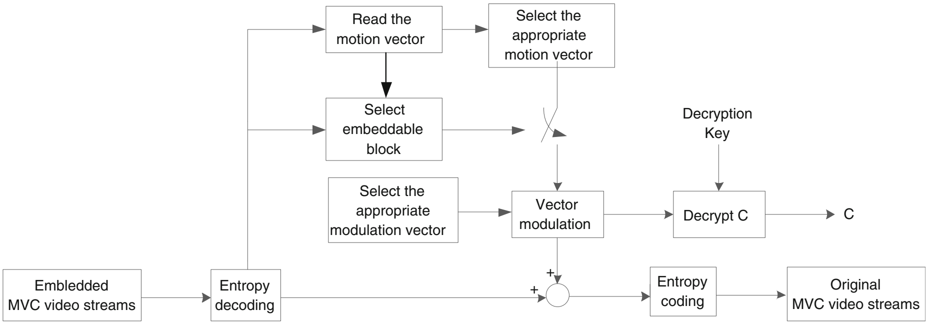

Fig. 6 Proposed data extraction scheme diagram

motion vector based on modulo modulation. Then, all the motion vectors are entropy coded to get the target original video.

3.2 The reversibility of the algorithm

In this section, we will demonstrate the reversibility of the proposed algorithm. According to **Algorithm 1**, we can draw that:

Case1:

$$\begin{aligned} &\text{If } 0 < \mathbf{M}_i\bullet\mathbf{\Delta} < \|\mathbf{\Delta}\|^2 \,\&\&\, C_i' = 1 \\ &\text{Then } \quad \mathbf{M}_i' = \mathbf{M}_i + \mathbf{\Delta} \\ &\qquad \mathbf{M}_i'\bullet\mathbf{\Delta} = (\mathbf{M}_i + \mathbf{\Delta})\bullet\mathbf{\Delta} = \mathbf{M}_i\bullet\mathbf{\Delta} + \|\mathbf{\Delta}\|^2 \\ &\qquad \mathbf{M}_i'\bullet\mathbf{\Delta} \in \left(\|\mathbf{\Delta}\|^2, 2\|\mathbf{\Delta}\|^2\right) \end{aligned} \tag{3}$$

Case2:

$$\begin{aligned} &\text{If } 0 < \mathbf{M}_i\bullet\mathbf{\Delta} < \|\mathbf{\Delta}\|^2 \&\& C_i' = 0 \\ &\text{Then } \quad \mathbf{M}_i' = \mathbf{M}_i \\ &\qquad \mathbf{M}_i' \bullet \mathbf{\Delta} \in \left(0, \|\mathbf{\Delta}\|^2\right) \end{aligned} \tag{4}$$

Case3:

$$\begin{aligned} &\text{If } -\|\mathbf{\Delta}\|^2 < \mathbf{M}_i\bullet\mathbf{\Delta} \le 0 \;\&\&\; C_i' = 1 \\ &\text{Then } \quad \mathbf{M}_i' = \mathbf{M}_i - \mathbf{\Delta} \\ &\qquad \mathbf{M}_i'\bullet\mathbf{\Delta} = (\mathbf{M}_i - \mathbf{\Delta})\bullet\mathbf{\Delta} = \mathbf{M}_i\bullet\mathbf{\Delta} - \|\mathbf{\Delta}\|^2 \\ &\qquad \mathbf{M}_i'\bullet\mathbf{\Delta} \in \left(-2\|\mathbf{\Delta}\|^2, -\|\mathbf{\Delta}\|^2\right] \end{aligned} \tag{5}$$

Case4:

$$\begin{aligned} &\text{If } -\|\mathbf{\Delta}\|^2 < \mathbf{M}_i\bullet\mathbf{\Delta} \le 0 \&\& C_i' = 0 \\ &\text{Then } \quad \mathbf{M}_i' = \mathbf{M}_i \\ &\qquad \mathbf{M}_i'\bullet\mathbf{\Delta} = \mathbf{M}_i\bullet\mathbf{\Delta} \\ &\qquad \mathbf{M}_i'\bullet\mathbf{\Delta} \in \left(-\|\mathbf{\Delta}\|^2, 0\right] \end{aligned} \tag{6}$$

Table 1 The inner product between $\mathbf{M}_i'$ and $\boldsymbol{\Delta}$

Original inner product	Embedding characters	Operating	Inner product after Embedding
$0<\mathbf{M}_i\bullet\boldsymbol{\Delta}<\|\boldsymbol{\Delta}\|^2$	$C_i'=0$	$\mathbf{M}_i'=\mathbf{M}_i$	$(0,\|\boldsymbol{\Delta}\|^2)$
	$C_i'=1$	$\mathbf{M}_i'=\mathbf{M}_i+\boldsymbol{\Delta}$	$(\|\boldsymbol{\Delta}\|^2,2\|\boldsymbol{\Delta}\|^2)$
$-\|\boldsymbol{\Delta}\|^2<\mathbf{M}_i\bullet\boldsymbol{\Delta}\leq 0$	$C_i'=0$	$\mathbf{M}_i'=\mathbf{M}_i$	$(-\|\boldsymbol{\Delta}\|^2,0]$
	$C_i'=1$	$\mathbf{M}_i'=\mathbf{M}_i-\boldsymbol{\Delta}$	$(-2\|\boldsymbol{\Delta}\|^2,-\|\boldsymbol{\Delta}\|^2]$
$\mathbf{M}_i\bullet\boldsymbol{\Delta}\geq\|\boldsymbol{\Delta}\|^2$	NULL	$\mathbf{M}_i'=\mathbf{M}_i+\boldsymbol{\Delta}$	$[2\|\boldsymbol{\Delta}\|^2,+\infty)$
$\mathbf{M}_i\bullet\boldsymbol{\Delta}\leq-\|\boldsymbol{\Delta}\|^2$	NULL	$\mathbf{M}_i'=\mathbf{M}_i-\boldsymbol{\Delta}$	$(-\infty,-2\|\boldsymbol{\Delta}\|^2]$

Case5:

$$\begin{array}{ll}\text{If} & \mathbf{M}_i\bullet\boldsymbol{\Delta}\geq\|\boldsymbol{\Delta}\|^2\\ \text{Then} & \mathbf{M}_i'=\mathbf{M}_i+\boldsymbol{\Delta}\\ & \mathbf{M}_i'\bullet\boldsymbol{\Delta}=(\mathbf{M}_i+\boldsymbol{\Delta})\bullet\boldsymbol{\Delta}=\mathbf{M}_i\bullet\boldsymbol{\Delta}+\|\boldsymbol{\Delta}\|^2\\ & \mathbf{M}_i'\bullet\boldsymbol{\Delta}\in\left[2\|\boldsymbol{\Delta}\|^2,+\infty\right)\end{array} \tag{7}$$

Case6:

$$\begin{array}{ll}\text{If} & \mathbf{M}_i\bullet\boldsymbol{\Delta}\leq-\|\boldsymbol{\Delta}\|^2\\ \text{Then} & \mathbf{M}_i'=\mathbf{M}_i-\boldsymbol{\Delta}\\ & \mathbf{M}_i'\bullet\boldsymbol{\Delta}=(\mathbf{M}_i-\boldsymbol{\Delta})\bullet\boldsymbol{\Delta}=\mathbf{M}_i\bullet\boldsymbol{\Delta}-\|\boldsymbol{\Delta}\|^2\\ & \mathbf{M}_i'\bullet\boldsymbol{\Delta}\in\left(-\infty,-2\|\boldsymbol{\Delta}\|^2\right]\end{array} \tag{8}$$

The inner product between the motion vector of the embedded data ($\mathbf{M}_i'$) and the modulation vector ($\boldsymbol{\Delta}$) is shown in Table 1:

If we embed data into the motion vector, a conclusion is draw n that:

$$\left\|\mathbf{M}_i'\bullet\boldsymbol{\Delta}\right\|<2\|\boldsymbol{\Delta}\|^2 \tag{9}$$

And we can use it as a condition when extracting data. More importantly, the process is reversible. In other words, the original MV ($\mathbf{M}_i$) can be restored according to the modified MV ($\mathbf{M}_i'$). For example, if we have the value of $\mathbf{M}_i'$, we can calculate $\mathbf{M}_i'\bullet\boldsymbol{\Delta}$, then,

Table 2 The motion vector and the embedded characters after extraction

Inner product after Embedding	Extracting characters	Operating	Restore
$(0,\|\boldsymbol{\Delta}\|^2)$	$C_i'=0$	$\mathbf{M}_i=\mathbf{M}_i'$	YES
$(\|\boldsymbol{\Delta}\|^2,2\|\boldsymbol{\Delta}\|^2)$	$C_i'=1$	$\mathbf{M}_i=\mathbf{M}_i'-\boldsymbol{\Delta}$	YES
$(-\|\boldsymbol{\Delta}\|^2,0]$	$C_i'=0$	$\mathbf{M}_i=\mathbf{M}_i'$	YES
$(-2\|\boldsymbol{\Delta}\|^2,-\|\boldsymbol{\Delta}\|^2]$	$C_i'=1$	$\mathbf{M}_i=\mathbf{M}_i'+\boldsymbol{\Delta}$	YES
$[2\|\boldsymbol{\Delta}\|^2,+\infty)$	NULL	$\mathbf{M}_i=\mathbf{M}_i'-\boldsymbol{\Delta}$	YES
$(-\infty,-2\|\boldsymbol{\Delta}\|^2]$	NULL	$\mathbf{M}_i=\mathbf{M}_i'+\boldsymbol{\Delta}$	YES

Case1:

$$\begin{array}{l}\text{If } \mathbf{M}_i' \bullet \boldsymbol{\Delta} \in \left(\|\boldsymbol{\Delta}\|^2, 2\|\boldsymbol{\Delta}\|^2 \right) \\ \text{Then } C_i' = 1 \&\& \mathbf{M}_i = \mathbf{M}_i' - \boldsymbol{\Delta}\end{array} \tag{10}$$

Case2:

$$\begin{array}{l}\text{If } \mathbf{M}_i' \bullet \boldsymbol{\Delta} \in \left(0, \|\boldsymbol{\Delta}\|^2 \right) \\ \text{Then } C_i' = 0 \&\& \mathbf{M}_i = \mathbf{M}_i'\end{array} \tag{11}$$

Case3:

$$\begin{array}{l}\text{If } \mathbf{M}_i' \bullet \boldsymbol{\Delta} \in \left(-2\|\boldsymbol{\Delta}\|^2, -\|\boldsymbol{\Delta}\|^2 \right] \\ \text{Then } C_i' = 1 \&\& \mathbf{M}_i = \mathbf{M}_i' + \boldsymbol{\Delta}\end{array} \tag{12}$$

Case4:

$$\begin{array}{l}\text{If } \mathbf{M}_i' \bullet \boldsymbol{\Delta} \in \left(-\|\boldsymbol{\Delta}\|^2, 0 \right] \\ \text{Then } C_i' = 0 \&\& \mathbf{M}_i = \mathbf{M}_i'\end{array} \tag{13}$$

Case5:

$$\begin{array}{l}\text{If } \mathbf{M}_i' \bullet \boldsymbol{\Delta} \in \left[2\|\boldsymbol{\Delta}\|^2, +\infty \right) \\ \text{Then } C_i' = NULL \&\& \mathbf{M}_i = \mathbf{M}_i' - \boldsymbol{\Delta}\end{array} \tag{14}$$

Case6:

$$\begin{array}{l}\text{If } \mathbf{M}_i' \bullet \boldsymbol{\Delta} \in \left(-\infty, -2\|\boldsymbol{\Delta}\|^2 \right] \\ \text{Then } C_i' = NULL \&\& \mathbf{M}_i = \mathbf{M}_i' + \boldsymbol{\Delta}\end{array} \tag{15}$$

The inner product between the motion vector of the embedded data ($\mathbf{M}_i'$) and the modulation vector ($\boldsymbol{\Delta}$) is shown in Table 2:

This indicates that the proposed algorithm is reversible. After extracting the data, we can recover the original data completely.

For example, if $Threshold=10, \boldsymbol{\Delta}=(1,-1)$, for a given motion vector $\mathbf{M}_i=(10,9)$, we can calculate that:

$$\|\mathbf{M}_i\| = \sqrt{10^2 + 9^2} > Threshold = 10 \tag{16}$$

Then

$$\begin{array}{l}\mathbf{M}_i \bullet \boldsymbol{\Delta} = (10, 9) \bullet (1, -1) = 1 \\ \|\boldsymbol{\Delta}\|^2 = (1, -1) \bullet (1, -1) = 2\end{array} \tag{17}$$

We have

$$0 < \mathbf{M}_i \bullet \boldsymbol{\Delta} < \|\boldsymbol{\Delta}\|^2 \tag{18}$$

If $C_i' = 0$ then $\mathbf{M}_i' = \mathbf{M}_i = (10,9)$
If $C_i' = 1$ then $\mathbf{M}_i' = \mathbf{M}_i + \boldsymbol{\Delta} = (10,9) + (1,-1) = (11,8)$

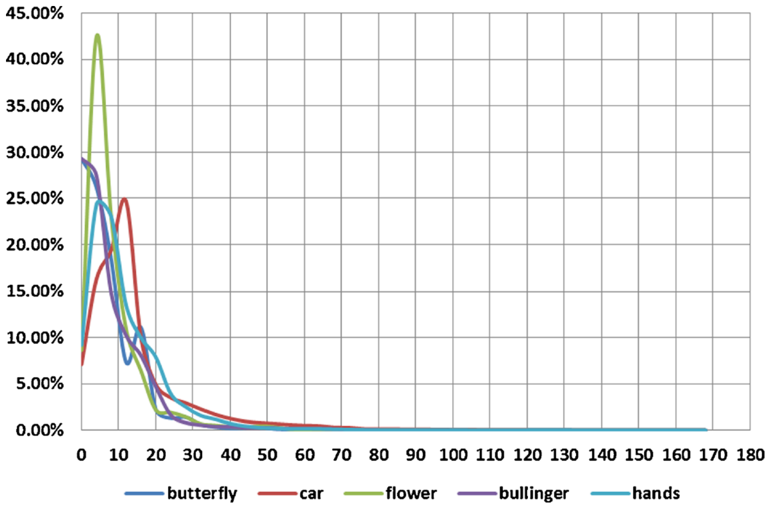

Fig. 7 The statistical properties of the motion vector in b4-frames (80 frames)

The embedding process ends and there is 1 bit data being embed into the motion vector according to **Algorithm 1**.

When we extract the hidden data, if we have $\mathbf{M}_i^{'}$=(11,8), then we can calculate that:

$$\left\|\mathbf{M}_i'\right\| = \sqrt{11^2 + 8^2} > Threshold = 10 \tag{19}$$

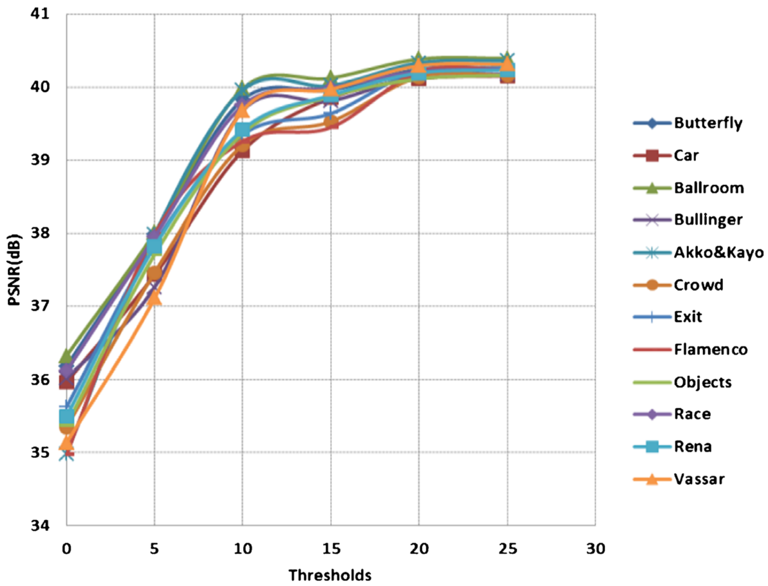

Fig. 8 The PSNR value under different thresholds (**Δ**=(1,−1))

Then

$$\begin{aligned} \mathbf{M}_i'\bullet\mathbf{\Delta} &= (11,8)\bullet(1,-1) = 3 \\ \|\mathbf{\Delta}\|^2 &= (1,-1)\bullet(1,-1) = 2 \end{aligned} \tag{20}$$

We obtain

$$\mathbf{M}_i'\bullet\mathbf{\Delta}\in\left(\|\mathbf{\Delta}\|^2, 2\|\mathbf{\Delta}\|^2\right)$$

Then

$$\begin{aligned} C_i' &= 1 \\ \mathbf{M}_i &= \mathbf{M}_i'-\mathbf{\Delta} = (11,8)-(1,-1) = (10,9) \end{aligned} \tag{21}$$

The extracting process ends and there is 1 bit data being extracted from the motion vector according to **Algorithm 2**. And the original motion vector is restored completely.

4 Experimental results

We implement the hiding and extraction by **Algorithm 1** and **Algorithm 2,** and integrate them to the MVC encoder and decoder. All implementations are experimented on a Lenovo compatible computer with an Intel Core 2 Due E5500 CPU 2.8 GHz and 2 GB RAM. The operating system is Microsoft Windows 7 SP1, the program development environment is Visual C++2010, and the implementation

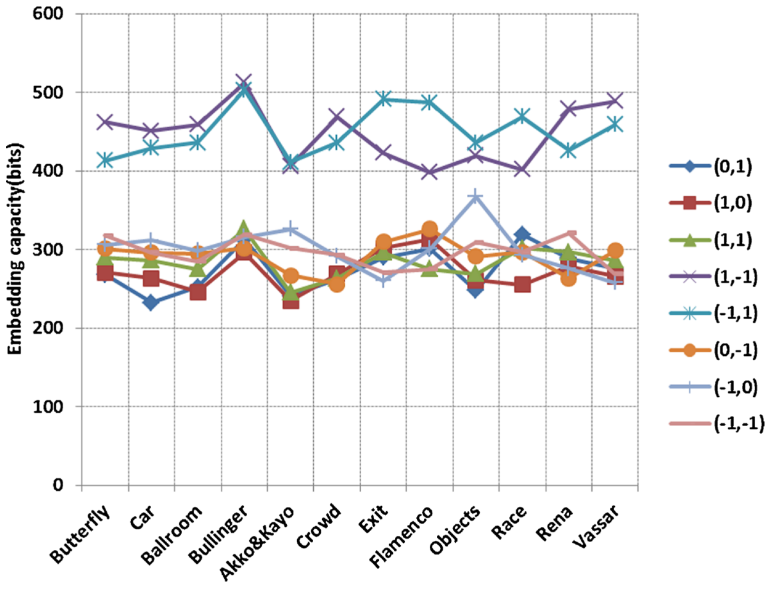

Fig. 9 The embedding capacity under different modulation vectors (*Threshold*=10)

platform is JM 18.4. As the message bits are embedded using MVs, the embedding capacity is measured by the average embedded bits per b4-frame. As shown in Fig. 8, 12 standard test sequences (Butterfly, Car, Ballroom, Bullinger, Akko&Kayo, Crowd, Exit, Flamenco, Objects, Race, Rena, Vassar) with different levels of motion in the 4:2:0 YUV format are used in our experiments, and they each have a frame size of 640×480 which corresponds to 1,200 MBs per frame. The size of GOP is chosen to be 8 and its frame structure is set as shown in Fig. 1. Just like most of the literature, we use PSNR, SSIM [32], embedding capacity, decoding time, and bitrate as the indicators to evaluate the proposed algorithm.

4.1 The statistical properties of the motion vector in b4-frames

Since the data will be embedded into the motion vector of each block of b4-frames, we first analyze the statistical properties of the motion vector. The distribution of the modulus of the motion vectors is shown in Fig. 7. It is obvious that most of the modulus is in a small range. This will help us select the optimal threshold to achieve a balance between the video quality and the embedded capacity.

Sequence	**Original**	**Embedded**
Butterfly		
Car		
Ballroom		
Bullinger		

Fig. 10 Visual effect of the original and embedded frame. (The first b4-frames, *Threshold*=10, $\boldsymbol{\Delta}=(1,-1)$)

Table 3 Embedding capacity comparison between the algorithms in [27, 30] and the proposed algorithm for QP=28. (bits/b4-frame)

Sequence	Algorithm in [30]	Algorithm in [27]	Proposed
Butterfly	234.23	213.39	462.36
Car	212.36	189.25	451.21
Ballroom	226.86	206.24	459.84
Bullinger	256.91	231.16	512.52
Akko&Kayo	186.21	165.59	413.93
Crowd	212.35	235.75	451.03
Exit	159.28	136.84	385.21
Flamenco	293.11	238.49	523.25
Objects	269.21	215.78	463.28
Race	198.38	160.36	416.24
Rena	241.15	203.92	429.19
Vassar	132.19	102.82	326.13

4.2 The relationship between the threshold and the video quality

The quality of the video within data embedded is mainly determined by the threshold we choose. The threshold also affects the embedding capacity. A smaller threshold may bring greater embedding capacity, however, it may also cause decline in the video quality. We have studied the PSNR of the video which hides data with the different thresholds, as shown in Fig. 8. We can find that the video quality gradually increases with the increase of the threshold value.

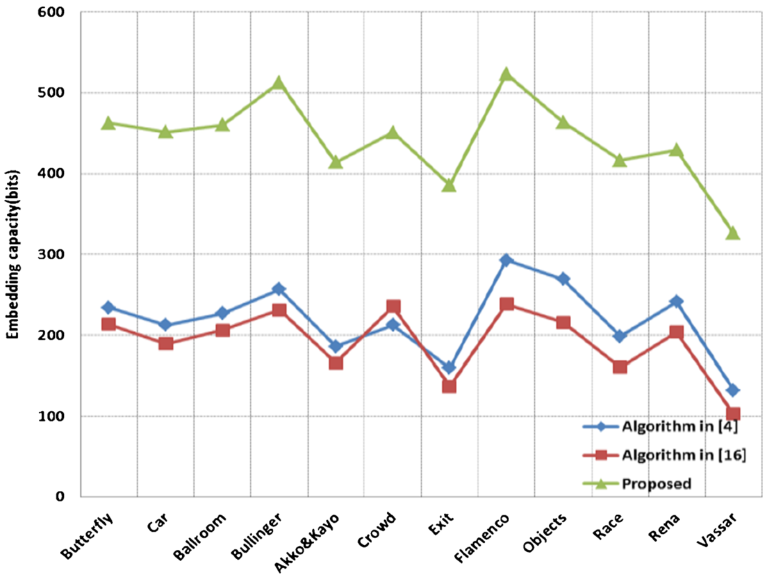

Fig. 11 Embedding capacity comparison between the algorithms in [27, 30] and the proposed algorithm for QP=28

Table 4 Bitrate increment rate comparison between the algorithms in [27, 30] and the proposed algorithm for QP=28

Sequence	Algorithm in [30]	Algorithm in [27]	Proposed
Butterfly	2.12 %	2.03 %	3.54 %
Car	1.86 %	1.79 %	2.13 %
Ballroom	2.03 %	1.93 %	2.36 %
Bullinger	2.23 %	2.06 %	3.52 %
Akko&Kayo	1.96 %	1.65 %	2.23 %
Crowd	2.26 %	2.09 %	3.19 %
Exit	1.69 %	1.51 %	2.11 %
Flamenco	2.67 %	2.18 %	3.63 %
Objects	2.55 %	2.21 %	3.53 %
Race	2.23 %	2.16 %	3.46 %
Rena	2.39 %	2.03 %	3.37 %
Vassar	1.64 %	1.09 %	2.38 %

4.3 The relationship between modulation vector and embedding capacity

In addition to the threshold value, the other parameter which affects the embedding capacity is modulation vector. Since the embedding conditions of the proposed method is that:

$$\|\mathbf{M}_i \bullet \boldsymbol{\Delta}\| < \|\boldsymbol{\Delta}\|^2$$

The embedding capacity is the average number of bits embedded into one b4-frame. It can be calculated by the following formula:

$$Capacity = \frac{Total\ \ of\ \ embledding\ \ bits}{Total\ \ of\ \ b4\text{-}frames} \tag{22}$$

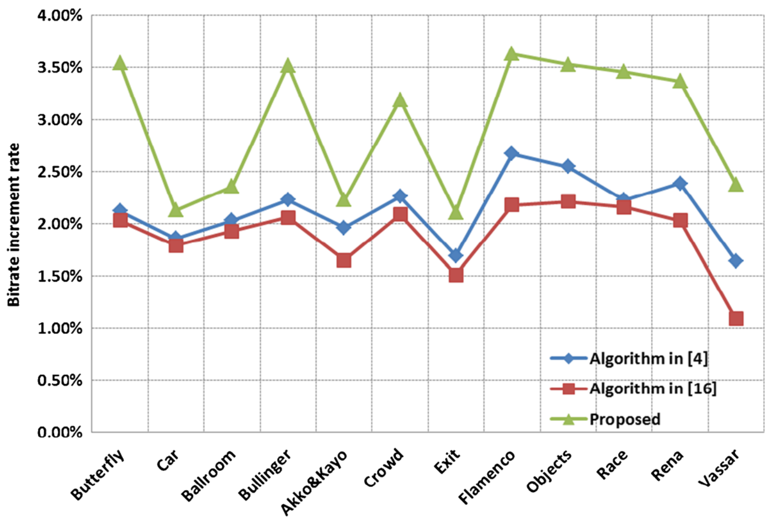

Fig. 12 Bitrate increment rate comparison between the algorithms in [27, 30] and the proposed algorithm for QP=28

Table 5 PSNR and SSIM comparison between the algorithms in [27, 30] and the proposed algorithm for QP=28

Sequence	Algorithm in [30]		Algorithm in [27]		Proposed	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Butterfly	39.15	0.987	39.26	0.990	39.10	0.986
Car	39.85	0.989	39.91	0.992	39.81	0.984
Ballroom	39.46	0.976	39.53	0.981	39.21	0.973
Bullinger	39.76	0.979	39.86	0.988	39.61	0.971
Akko&Kayo	39.88	0.990	39.92	0.993	39.73	0.980
Crowd	39.72	0.972	39.83	0.986	39.60	0.969
Exit	39.89	0.992	39.93	0.995	39.82	0.986
Flamenco	39.35	0.962	39.53	0.980	39.02	0.952
Objects	39.39	0.966	39.62	0.983	39.26	0.961
Race	39.68	0.979	39.76	0.987	39.56	0.972
Rena	39.55	0.964	39.69	0.984	39.66	0.960
Vassar	39.93	0.993	39.95	0.996	39.91	0.991

So, the selection of modulation vector directly determines the size of the embedding capacity. In order to maintain the quality of the videos, the values of modulation vector should be in a small range. The embedding capacity with the different modulation of vectors is also studied as shown in Fig. 9.

4.4 Data hiding performance

Through the study of the relationship between the threshold and the video quality in Section 4.2, we find that a smaller threshold may bring greater embedding capacity, however, it may cause

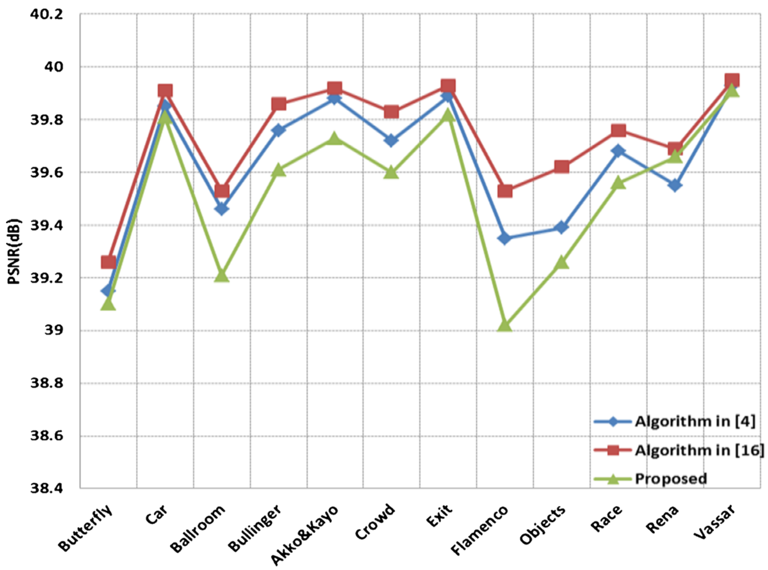

Fig. 13 PSNR comparison between the algorithm in [27, 30] and the proposed algorithm for QP=28

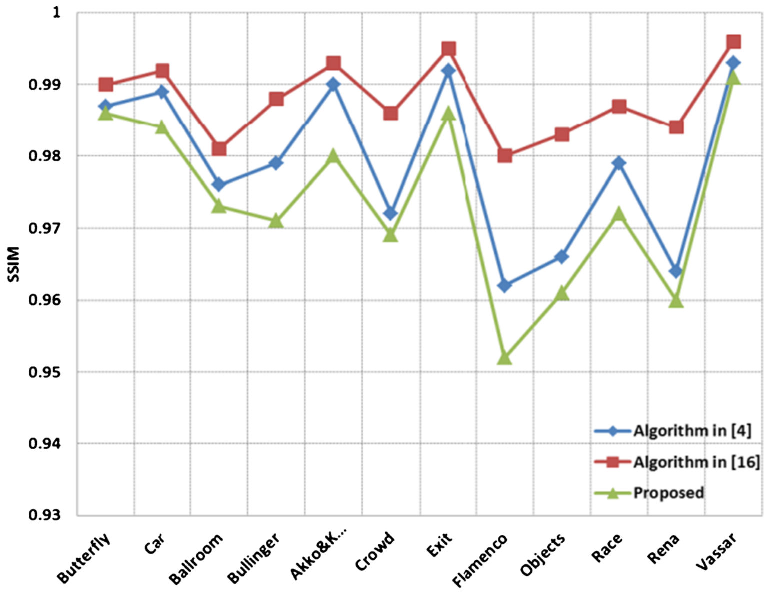

Fig. 14 SSIM comparison between the algorithms in [27, 30] and the proposed algorithm for QP=28

decline in the video quality. When *Threshold*<10, the PSNR value almost increases linearly as the threshold value increases. And when *Threshold*≥10, the increase in PSNR values becomes very gentle as the threshold value increases. So we take *Threshold*=10 as the lower bound of the threshold.

On the other hand, according to the distribution of the motion vector shown in Fig. 5, we can find that most of the modulus values are in a small range. If *Threshold*>20, there are less than 5 % of the motion vectors meeting the condition, which greatly restrict the embedding capacity. Therefore, we generally limit the threshold in the interval [13, 22].

Given the appropriate threshold and the modulation vector, such as *Threshold*=10, $\mathbf{\Delta}$=(1,−1), we implement the proposed algorithm on four standard test sequences. The size of GOP is chosen

Table 6 Decoding time comparison between the algorithms in [27, 30] and the proposed algorithm in terms of milliseconds per frame.(QP=28)

Sequence	Algorithm in [30]	Algorithm in [27]	Proposed
Butterfly	18.612	18.601	18.595
Car	18.614	18.611	18.592
Ballroom	18.621	18.616	18.596
Bullinger	18.625	18.620	18.597
Akko&Kayo	18.636	18.623	18.012
Crowd	18.703	18.692	18.603
Exit	18.233	18.210	18.013
Flamenco	19.012	19.001	18.899
Objects	19.126	19.023	18.901
Race	18.263	18.210	18.026
Rena	19.210	19.056	18.923
Vassar	18.012	17.986	17.832

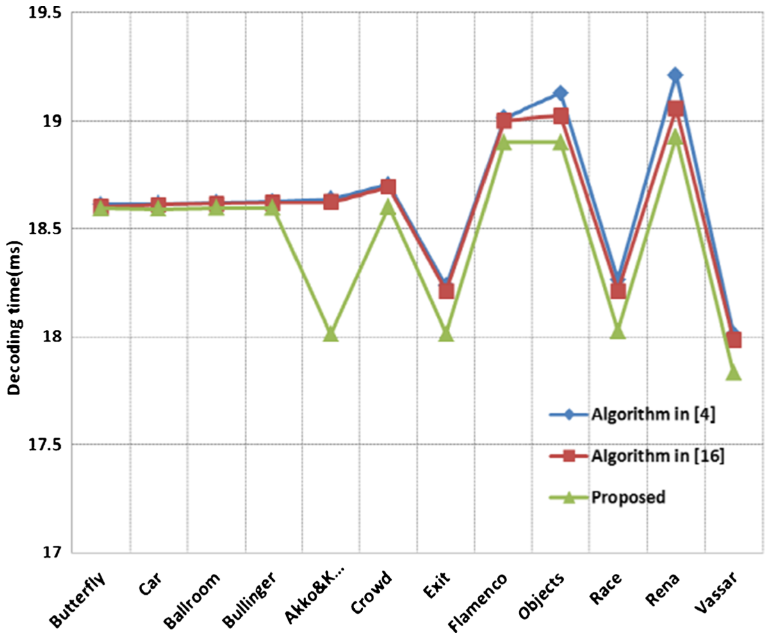

Fig. 15 Decoding time comparison between the algorithms in [27, 30] and the proposed algorithm in terms of milliseconds per frame (QP=28)

to be 8 and its frame structure is set as shown in Fig. 1. The data hiding performances of our algorithm are shown in Fig. 10.

Table 3 and Fig. 11 depict the embedding capacity of these test sequences when QP is set to 28. The embedding capacity improvement ratio ranges from 51 % to 93 % since the selected value of MVs are dependent on the content of test sequences. This is mainly because our algorithms take full advantage of the MVs of each block to hide data instead of DCT coefficients. And the embedding condition of our method is more lenient than the algorithms in [27, 30].

Table 4 and Fig. 12 illustrate the bitrate increment rate for these test sequences when QP is set to 28. The average bitrate increment rates of the algorithms in [27, 30] and the proposed algorithm are 2.15 %, 1.91 % and 2.97 %, respectively, and it indicates that the bitrate degradation is rather small.

Table 5 and Figs. 13 and 14 illustrate the quality comparison for both algorithms. Besides using PSNR to measure the quality of the embedded sequences, the SSIM index is employed to measure the visual quality [17]. When compared with the algorithms in [27, 30], although the PSNR degradation of the proposed algorithm are 0.11dBs and 0.21dBs, the SSIM index is close to that of the algorithms in [27, 30]. The similar SSIM indexes of the three concerned algorithms imply that the proposed algorithm improve the embedding capacity of the algorithms in [27, 30] without visual quality degradation.

The decoding time performances of the three algorithms are shown in Table 6 and Fig. 15. We can find that each algorithm requires 18.68 ms, 18.64 ms and 18.50 ms to decode each frame from the compressed video sequence. Table 6 also reveals that the decoding time performance of each algorithm satisfies real-time requirement.

4.5 Robustness analysis and discussions

At present the mainly attacks of steganography are the geometric attacks and the physical attacks. When encountering the common geometric attacks, such as cutting, deformation and other operations, the algorithm represents a lower robustness. However, such an operation would cause a greater impact on the video quality, and a serious decline in the quality of the video is generally not acceptable by the recipient. Therefore, the proposed algorithm is most likely to suffer from the physical attacks, such as noise pollution, frame loss and so on. The proposed algorithm has a better robustness on regular physical attacks. The main reasons are described below.

First, being as the control information, the values of MV have the higher priority in transmission [12]. If modifying or lost more control information, it will lead to a serious decline in the video quality, and the video even cannot be decoded. Thus, during the transmission, the motion vector is much better protected. When encountering the common attacks, such as error, noise and so on, the value of MV is relatively robust.

Second, the embedding medium is 3D videos. There are very large spaces to embed data into 3D videos, so we can repeat the embedding process in order to effectively solve the problem of missing frames. Through the study in Section 2, we obtain that there are four b4-frames in a GOP (Length=8, Total frames=16 with two viewpoints). This means that one quarter frames are b4-frames in the 3D MVC videos. Furthermore, the idea of random sequence can also be introduced to further improve the undetectability and the randomness of the algorithm. For instance, we can add three random sequences to select the frames, sub-MBs and MVs for hiding data respectively. And it will effectively solve the problem of missing frames, blocks and so on.

Third, in order to further improve the robustness of the algorithm, we can also make use of the classical theory of error correction coding, such as BCH and Convolutional Codes, to preprocess the information. It is the work in progress and we will continue to make deep research in this direction.

In addition, this paper focuses on reversibility, so we did not do in-depth analysis of the robustness. We will use robust methods (e.g., secret sharing and error correction coding), to make our algorithms more robust to the variety of attacks in the future work.

5 Conclusions

We have presented a new data hiding algorithm to achieve reversible steganography for MVC video. The contribution of this work is threefold. Firstly, unlike most of the existing algorithms, we hide data into the motion vector of b4-frames. This is more suitable for the 3D MVC video with the structure of hierarchical B-frames. Secondly, establishing the inner product between the motion vector and the modulation vector to achieve the reversibly steganography is a new attempt. Thirdly, the proposed algorithm takes 3D videos as the carrier. As far as we know, this is the first reversible steganography algorithm without inter distortion drift for 3D video. Based on twelve video test sequences, experimental results demonstrate the large embedding capacity and low complexity of the proposed algorithm while keeping human visual effect unchanged in terms of PSNR and SSIM index.

Acknowledgments The authors would sincerely like to thank the anonymous reviewers of the paper for several insightful comments. They also thank the editor Ms. Angie Malanday for her efforts in revising the paper. The work described in this paper was supported by the National Natural Science Foundation of China under Grant (Name: Research on Steganography for 3D H.264 Video Streams without Intra-frame Distortion Drift. No: 61272407).

References

1. Alattar AM (2004) Reversible watermark using the difference expansion of a generalized integer transform [J]. IEEE Transactions on Image Processing 13(8):1147–115
2. Aly H (2011) Data hiding in motion vectors of compressed video based on their associated prediction error. IEEE Trans Inf Forensics Secur 6(1):14–18
3. Barton JM, Method and apparatus for embedding authentication information within digital data. 1997, US Patent 6,115,818.
4. Bender W, Gruhl D, Morimoto N, Lu A (1996) Techniques for data hiding. IBM Syst J 35(no. 3, 4):313–336
5. Cao Y, Zhao X, Feng D (2012) Video steganalysis exploiting motion vector reversion-based features. IEEE Signal Process Lett 19(1):35–38
6. Celik MU, Sharma G, Tekalp AM, Saber E (2002) Reversible data hiding. Proc IEEE Int Conf Image Process 2:157–160
7. Chang CC, Lu TC (2006) A difference expansion oriented data hiding scheme for restoring the original host images [J]. The Journal of Systems & Software 79(12):1754–1766
8. Chang CC, Tai WL, Lin CC (2006) A reversible data hiding scheme based on side-match vector quantization. IEEE Trans Circuits Syst Video Technol 16(10):1301–1308
9. Fallahpour M, Megías D (2009) Reversible Data Hiding Based on H.264/AVC Intra Prediction. Lecture Notes in Computer Science, no.5450. Springer, Berlin, pp 52–60
10. Fallahpour M, Megias D, Ghanbari M (2011) Reversible and high-capacity data hiding in medical images. IET Image Processing 5(2):190–197
11. Fang D, Chang L (2006) Data hiding for digital video with phase of motion vector. Proc. Int. Symposium on Circuit and Systems (ISCAS)[C] 1422–1425.
12. Ho YS, Oh KJ. Overview of multi-view video coding. Proc. 14th Int. Workshop Syst. Signals Image Process., 6th EURASIP Conf. Focused Speech Image Process., Multimedia Commun. Services, pp.5–12 2007.
13. Hong W, Chen TS, Chang YP, Shiu CW (2010) A high capacity reversible data hiding scheme using orthogonal projection and prediction error modification. Signal Process 90(11):2911–2922
14. Hsien-Wen T, Chi-Chen C (2008) An extended difference expansion algorithm for reversible Watermarking[J]. Image and Vision Computing 26(8):1148–1153
15. Hu Y, Lee H-K, Li J (2009) DE-based reversible data hiding with improved overflow location map. IEEE Trans Circuits Syst Video Technol 19(2):250–260
16. Hwang J, Kim JW, Choi JU (2006) A reversible watermarking based on histogram shifting, Int. Workshop on Digital Watermarking, Lecture Notes in Computer Science 4283:348–361
17. ITU-T Recommendation H.264 and ISO/IEC 14496–10 AVC (2010) Advanced Video Coding for Generic Audiovisual Services
18. Jing H, He X, Han Q, Niu X (2012) Motion vector based information hiding algorithm for H. 264/AVC against motion vector steganalysis. Intelligent Information and Database Systems Lecture Notes in Computer Science 7197:91–98
19. Kim S, Hong Y, Won C (2007) Data hiding on H.264/AVC compressed video. Image Anal Recog 4633(2007):698–707
20. Kung CH, Jeng JH, Lee YC, Hsiao HH, Cheng WS. Video Watermarking Using Motion Vector. 16th IPPR Conference on computer vision, graphics and image processing, 2003: 547–551.
21. Kutter, M., Jordan, F., Ebrahimi, T.: Proposal of a watermarking technique for hiding/retrieving data in compressed and decompressed video. Technical report M2281, ISO/IEC document, JTC1/SC29/WG11 (1997).
22. Lie WN, Lin CI, Tsai DC, Lin GS (2005) Error resilient coding based on reversible data embedding technique for H.264/AVC video. Proc. IEEE Int. Conf. Multimedia and Expo 1174–1177
23. Merkle P, Smolic A, Mueller K, Wiegand T (2007) Efficient prediction structures for multiview video coding. IEEE Trans Circuits Syst Video Technol 17(11):1461–1473

24. Ni Z et al (2006) Reversible Data Hiding. IEEE Trans Circuits Syst Video Technol 16(3):354–362
25. Noorkami M, Mersereau RM (2007) A framework for robust watermarking of H.264-encoded video with controllable detection performance. IEEE Trans Inform Forensics Security 2(1):14–23
26. Profrock D, Richter H, Schlauweg M, Muller E (2005) H.264-AVC video authentication using skipped macroblocks for an erasable watermark. Proc SPIE Visual Commun Image Process 5960: 1480–1489
27. Qin C, Chang CC, Huang YH, Liao LT (2012) An Inpainting-Assisted Reversible Steganographic Scheme Using Histogram Shifting Mechanism. IEEE Transactions on Circuits and Systems for Video Technology, (99): 1–11
28. Thodi DM, Rodriguez JJ (2007) Expansion embedding techniques for reversible watermarking [J]. IEEE Tran On Image Processing 16(3):721–730
29. Thodi D M, Rodriguez J J. Reversible watermarking by Prediction-error expansion[C], IEEE Southwest Symposium on Image Analysis and Interpretation, Arizona, USA, 2004:21-25
30. Tian J (2003) Reversible data embedding using a difference expansion. IEEE Trans Circuits Syst Video Technol 13(8):890–896
31. Tsai P, Hu YC, Yeh HL (2009) Reversible image hiding scheme using predictive coding and histogram shifting. Signal Process 89:1129–1143
32. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: From error measurement to structural similarity. IEEE Trans Image Process 13(4):600–612
33. Yang CY, Lin CH, Hu WC (2011) Reversible data hiding by adaptive IWT-coefficient adjustment. Journal of Information Hiding and Multimedia Signal Processing 2(1):24–32
34. Zhang J, Ho ATS, Qiu G (2007) Robust video watermarking of H.264/AVC. IEEE Trans Circuits Syst II: Express Briefs 54(2):205–209
35. Zhang J, Li J, Zhang L (2001) Video watermark technique in motion vector. Proc. XIV Symp. Computer Graphics and Image Processing 179–182
36. Zhao Z, Yu N, Li X (2003) A novel video watermarking scheme in compression domain based on fast motion estimation. In: International Conference on Communication Technology 2:1878–1882

Guanghua Song was born in Heze, China in 1981, and received his B.S. degree from Yantai University, Yantai, China, 2004 and M.E. degree from Huazhong University of Science and Technology, Wuhan, China, 2007.

He is currently a PhD student in the Department of Computer Science and Technology at Huazhong University of Science and Technology. His research interests include network security and multimedia security.

Zhitang Li was born in Jianli, China in 1951, and received his M.E. degree in Computer Architecture from Huazhong University of Science and Technology, Wuhan, China, 1987, and PhD degree in Computer Architecture from Huazhong University of Science and Technology, Wuhan, China, 1992. His research interests include computer architecture, network security, and P2P networks.

He was the director of China Education and Research Network (CERNET) in Central China. He was a vice president of Department of Computer Science and Technology, Huazhong University of Science and Technology, China. He has published more than one hundred papers in the areas of network security, computer architecture, and P2P networks.

Juan Zhao was born in Xinxiang, China in 1985, and received her B.S. and M.E. degrees from Henan Normal University, Xinxiang, China, in 2007 and 2010.

She is currently a PhD student in the Department of Computer Science and Technology at Huazhong University of Science and Technology. Her research interests include network security and multimedia security.

Jun Hu was born in Wuhan, China in 1987 and received his B.S. degrees in computer science and technology from Huazhong University of Science and Technology, Wuhan, China, in 2010. His research interests include: computer and network security and data hiding in H.264/AVC streams.

He is currently working toward a Master degree with Huazhong University of Science and Technology.

Hao Tu was born in Wuhan, China in 1977, and received his B.S. and PhD degrees in computer science and technology from Huazhong University of Science and Technology, Wuhan, China, in 1999 and 2008.

His research interests include: computer and network security and multimedia forensics. He is currently a lecturer in the Network Center, Huazhong University of Science and Technology

.