ISSN 2354-0575

# AN ALGORITHM FOR REVERSIBLE DATA HIDING IN H.264/AVC

**Dinh-Chien Nguyen, Minh Chuan Pham, Thi Phuong Tran, Khanh Trinh Nguyen**
*Hung Yen University of Technology and Education*

Received: 20/01/2020
Revised: 10/02/2020
Accepted for publication: 15/02/2020

**Abstract:**

*Reversible data hiding is a technique for embedding secret data in a host, such as image, database, audio, and video, but it can recover the original host. By the histogram shifting technique, in this paper, a reversible data hiding in H.264/AVC is proposed with a purpose that the embedding capacity can achieve as higher as possible, simultaneously, the video can recover to the original better possible. This study can also prevent distortion drift. The experimental results show that the proposed algorithm can approximately recover to the original video. By comparing with the other studies, the proposed study further improves the embedding capacity, and can recover to the original video. A disadvantage of the algorithm is that it cannot correct the error bits for network attacks. So, in the future, we will use BCH code technique for robustness of data hiding with the proposed algorithm.*

***Keywords:*** *Reversible data hiding, DCT, H.264/AVC, embedding capacity, distortion drift, histogram shifting.*

## 1. Introduction

Cryptography is usually used for secured communication with the presence of third parties. To prevent third parties or the public could read the private messages, many studies have been exploring in steganography for vast of the host, such as image, audio, video, source code, database, DNA sequence, etc. The steganography is the art of concealing information in preventing detection.

Many steganography schemes [1-3] for digital media have been proposed in a few years. The video, is one of the various types of digital media, is usually used for steganography schemes because of its wide applications in both portable storage devices and Internet, such as surveillance camera and Youtube channels. In order to save storage space, the H.264/AVC (Advanced Video Coding), which was introduced from 2003 by I.E.G. Richardson [4], is usually applied to compress the video sequences.

H.264/AVC is a interested host for data hiding[5,6]. DCT coefficients of I-frames was used in all state-of-the-art schemes for embedding data into video sequences. However, these schemes suffered the intra-frame distortion drift issue. In order to solve this problem, Ma et al. [5] proposed a novel DCT-based steganography algorithm by selecting three quantized DCT paired-coefficients for carrying the secret data. Nevertheless, their scheme was the low visual quality of embedded video sequences and obtained unsatisfied embedding capacity. To improve the performance of Ma et al.'s scheme [5], in 2013, Lin et al. [6] classified the luminance block by five cases that explored the characteristic of the quantized DCT coefficients for data embedding. Even though, Lin et al.'s scheme has further gained the embedding capacity of Ma et al.'s scheme up to 0.15 bit per pixel (bpp), however, the embedding capacity unsatisfactory when the average embedding rate was smaller than 0.7 bpp.

To recover data from embedded images, in 2006, Ni et al. [7] proposed the method of the histogram shifting for still image. The method utilized the zero-point values and peak-point values of the histogram of an image. The study could embed more data than many of the existing reversible data hiding methods, but the Peak Signal-to-Noise Ratio (PSNR) always is 48.2dB for all kind of image. Based on histogram shifting technique, we first generate the histogram of the paired-coefficient values for three cases. After that, we find the zero-point value and peak-point value, and then shift the

histogram to the right hand. The secret data will be embedded into DCT coefficients that are peak-point values. To recover original video, after extraction hidden data, we only use histogram shifting back.

The rest of the paper is organized as follows. Some information about intra-frame prediction, embedding schedule analysis, and histogram shifting are introduced in section 2. Section 3 presents the proposed reversible data hiding scheme. The experimental results are shown in section 4. Conclusions of this paper are drawn in section 5.

## 2. Related works

### 2.1. Intra-frame prediction

In order to reduce the redundancy of Intra-frames, the intra prediction algorithm is used in H.264/AVC [4]. In the intra-frames, the blocks can be formed by 4×4 or 16×16 macroblocks. Since the human eyes are very sensitive to any modification of luminance values in 16×16 intra MBs, many studies have used 4×4 intra blocks to embed the secret data. Consider that the 16 samples, from *a* to *p* of the current block in Figure 1, are calculated based on the boundary pixels of the left and upper blocks, labeled from *A* to *M*. The left and upper blocks are used to predict the current block.

| M | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| I | a | b | c | d | | | | |
| J | e | f | g | h | | | | |
| K | i | j | k | l | | | | |
| L | m | n | o | p | | | | |

*Figure 1. The current luminance block* $\widehat{B_{i,j}}$

To prevent Intra-frame distortion drift, in 2010, Ma et al. [5] introduced the method for determining the 4×4 block conditions, which are Cond 1, Cond 2 and Cond 3, shown in Table 1.

*Table 1: Three conditions of the selected modes and its corresponding reference pixels*

| | **Mode name** | **Mode value** | **Reference pixels** |
|---|---|---|---|
| **Cond 1** | Right-Mode | 0, 3, or 7 | *d, h, l, p* |
| **Cond 2** | Under-Left-Mode & Under-Mode | 1 or 8 and 0, 1, 2, 4, 5, 6, or 8 | *m, n, o, p* |
| **Cond 3** | Under-Right-Mode | 0, 1, 2, 3, 7, or 8 | *p* |

To further improve embedding capacity of Ma et al.'s scheme, in 2013, Lin et al. [6] fully exploited the remaining 54% luminance blocks, and improved the data hiding capacity. In this study, the authors defined five categories, named Cat1, Cat2, Cat3, Cat4, and Cat5. According to methods in [5] and [6], three cases are launched in this study, shown in Table 2.

*Table 2: Three cases for prediction modes of the block*

| **Cases** | **Cond 1** | **Cond 2** | **Cond 3** | **Reference pixels** |
|---|---|---|---|---|
| ***Case1*** | True | False | X | *d, h, l, p* |
| ***Case2*** | False | True | X | *m, n, o, p* |
| ***Case3*** | False | False | True | *p* |

X – *Do not care;*

Since the embedding capacity of this study seems high, the three cases are used. When we need more secret data are embedded into videos, the remaining categories in [6] can be discovered.

### 2.2. Embedding procedure analysis

Integer cosine transform (ICT), a kind of Discrete Cosine Transform (DCT), is usually used in H.264/AVC standard. Since the human eyes are less sensitive to the brightness, we only use 4×4 luminance blocks to embed data, and apply the ICT transform for 4×4 blocks, shown in (1).

$$W = C_f R C_f^T \tag{1}$$

Where W is the matrix of undetermined DCT coefficients corresponding to the residual block $R_{4\times4}$; $C_f^T$ is transformed matrix of $C_f$, and

$$C_f = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

With $qbits = 2^{15+floor\left(\frac{QP}{6}\right)}$, and

$$PF = \begin{bmatrix} a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \\ a^2 & ab/2 & a^2 & ab/2 \\ ab/2 & b^2/4 & ab/2 & b^2/4 \end{bmatrix},$$

$$a = 1/2, b = \sqrt{2/5},$$

We can calculate the basic quantization as the following equation,

$$\hat{W} = round\left(\frac{W.PF}{Qstep}\right) \tag{2}$$

*Qstep* is the quantizer step size, which is determined by quantization parameter (QP), and the factor (*PF/Qstep*) can be implemented in the reference model software as a multiplication by a

factor *MF* and right-shift, we have

$$\frac{MF}{2^{15+floor\left(\frac{QP}{6}\right)}} = \frac{PF}{Qstep}$$

The secret data is embedded into the quantized luminance DCT coefficients as in following formula,

$$\widehat{W'} = \widehat{W} + \Delta \quad (3)$$

where $\Delta = (a_{i,j})_{4\times4}$ is the 4×4 error matrix added to the 4×4 quantized DCT coefficient matrix $\widehat{W}$ by data hiding.

### 2.3. Histogram Shifting

Ni et al, 2006 [7] had generated the grayscale image's (512 × 512 × 8) histograms. In this histogram, the zero point and the peak point have found by corresponding to the grayscale value. The zero point means no pixel in the given image, and the peak point is the maximum number of pixel in the given image. The finding of peak point was proposed, in order to increase the embedding capacity as large as possible.

## 3. The proposed reversible data hiding scheme

### 3.1. Histogram generation and shifting

In this study, the histogram based on the paired-coefficients values is generated. First, the modes of macroblocks are predicted, and only allow all macroblocks which are in *Case 1*, *Case 2* and *Case 3*. After that, the histogram will be generated by coefficients values. The peak point can be predicted by finding maximum value in the histogram. The zero point is easily predicted by scanning from peak point value to a value in the histogram that is zero to the right or to the left. Finally, the histogram shifting is performed. In order to easy know, we consider that the coefficients in macroblocks are A on column 1 with *Case 1*, and on row 1 with *Case 2* and *Case 3*, and the coefficients in macroblocks are B on column 3 with *Case 1*, and on row 3 with *Case 2* and *Case 3* (Figure 2).

*Case 1*

| | | | |
|---|---|---|---|
| $A_1$ | | $B_1$ | |
| $A_2$ | | $B_2$ | |
| $A_3$ | | $B_3$ | |
| $A_4$ | | $B_4$ | |

*Case 2* and *Case 3*

| | | | |
|---|---|---|---|
| $A_1$ | $A_2$ | $A_3$ | $A_4$ |
| | | | |
| $B_1$ | $B_2$ | $B_3$ | $B_4$ |
| | | | |

*Figure 2. A is row (column) 1, and B is row (column) 3*

Because of the changing more coefficients will affect the video quality, we see that the right values are larger than the left values in the histogram, therefore, we increase the right values, from peak_point + 1. The histogram shifting is performed by following formula,

$$[A_i, B_i] = [A_i + 1, B_i - 1] \quad \text{if } (A_i \geq \text{peak\_point}+1) \quad (4)$$

When the Cases meet in 1, 2 and 4, the paired-coefficient values was checked. If the coefficient $A_i$ ($i$=1…4) equals to or greater than 1, increase it one value. To avoid drift distortion, we must keep the balance of paired-coefficient value. So that, if $A_i$ is increased, the $B_i$ should be decreased, and vice versa.

The histogram shifting phase is illustrated by the following algorithm,

**Histogram shifting phase**

**Input:** Macroblocks, binary secret data (b)

**Output:** Macroblock with new value of paired-coefficients

**Step 1:** Load Case classification table, which contains Macroblocks' case.

**Step 2:** If Macroblock is in Case 1, Case 2 and Case 3, apply formula (9) to shift the histogram.

Because of the peak_point values in all of ten videos, which are used in this study, are zero, therefore, zero is considered the pick_point value. Figure 3 shows an illustration of the histogram shifting procedure. All values of *A* greater than or equal to 1 are increased by 1. In order to avoid distortion drift, all values of corresponding *B* are decreased by 1. After shifting, all values 1 of *A* do not exist, and the data can embed on all of values *A* that equal to zero.

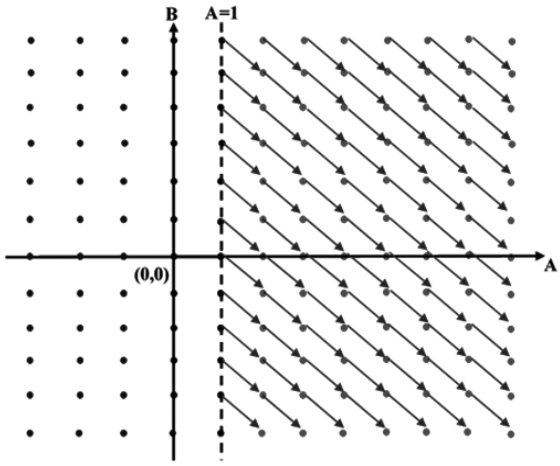


*Figure 3. Illustration for the histogram shifting procedure*

The procedure for embedding secret data shows in the next section.

### 3.2. Embedding process

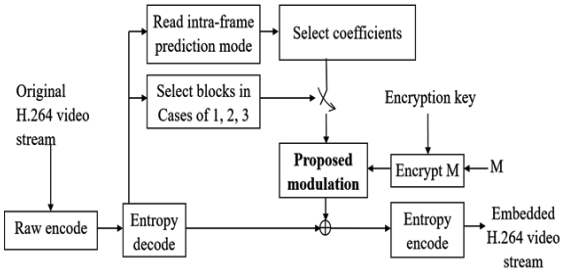


*Figure 4. The diagram of the embedding process.*

Figure 4 shows that the raw videos sequences have been decoded to the frames, contain I-frames, P-frames, and B-frames. In order to ensure video quality, we only perform with the I-frames. After entropy encoding, the I-frames are read to predict modes, and select macroblocks. Because the values from peak_point +1 have shifted to the right hand, we can embed secret bits into coefficient that its value is equal to peak point value. In this study, we generate the histogram of *A*'s coefficients.

Assume that, the secret data bit is s and coefficient is $A_i$; $A_i$ is $Y_{I,i}$, $i$ =1,..,4 with MBs in *Case 1*, and $Y_{i,1}$, $i$ =1…4, with MBs in *Case 2*. *Case 3* can handle the same with *Case 2*. The proposed modulation operates our embedding scheme. The secret data *s* is embedded into macroblocks of the frames by the following formula,

$$[A_i, B_i] = \begin{cases} [A_i + 1, B_i - 1] & \text{if (s = 1), } A_i = 0; \\ [A_i, B_i] & \text{if (s = 0), } A_i = 0 \end{cases} \quad (5)$$

When the Cases meet in 1, 2 and 4, the paired-coefficient values was checked. If the coefficient $A_i$ equals to 0, peak point value, it can be increased when the secret bit is 1, otherwise, the coefficient $A_i$ cannot be changed. To avoid drift distortion, we have to keep the balance of paired-coefficient value. So that, if $A_i$ is increased, the $B_i$ should be decreased, and vice versa. The embedding phase is illustrated by the following algorithm,

**Embedding algorithm**

**Input:** blocks, binary secret(b)
**Output:** Embedded blocks
**Step 1:** Load *Case classification* table, which contains blocks' case.
**Step 2:** If block is in Case 1, Case 2 and Case 3, we embed secret data into coefficients by formula (5)

Entropy encode module will generate the video bitstream, which includes frames and the embedded data. The bitstream will be transferred to the receiver, and then will be processed by extraction and recovering process.

### 3.3. Extraction and recovering process

The Embedded H.264 video stream in Figure 5 is entropy encoded to macroblocks. Macroblocks are then selected to extract the hidden data. The hidden data H ($h_1$, $h_2$,.., $h_n \in \{0,1\}$) is extracted by following formula,

$$h_j = \begin{cases} 1 & \text{if } A_i = 1; \\ 0 & \text{if } A_i = \text{peak\_point}; \end{cases} \quad (6)$$

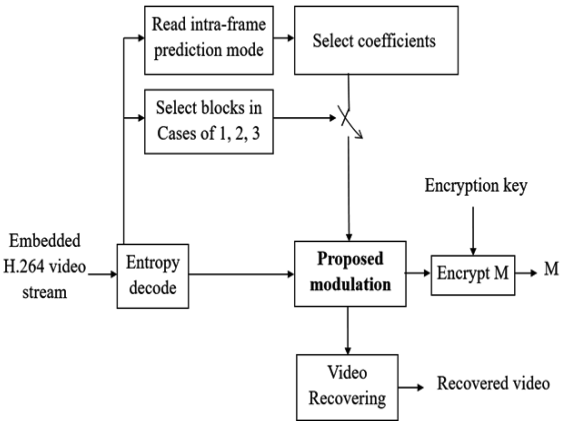


*Figure 5. The diagram of extraction and recovering process.*

Since the embedded data bit '1' was contained in coefficients that are peak_point + 1, and embedded data bit '0' was contained in the coefficient values are peak_point, we can extract data by checking coefficient values. If coefficient values are peak_point, peak_point + 1, the hidden data bit $h_j$ equal to 0, 1, respectively.

After extract the embedded data, the coefficient values should be recovered.

$$[A_i, B_i] = \begin{cases} [A_i - 1, B_i + 1] & \text{if } A_i = 1; \\ [A_i, B_i] & \text{if } A_i = 0 \end{cases} \quad (7)$$

By the same way with extraction process, the original value of coefficients can recover by reducing coefficient values that are 1. The following algorithm illustrates the extraction and recovering process,

**Extraction and recovering algorithm**

```
Input: EMD array (E); blocks
Output: hiding data
Step 1: Load Case classification table, which contains blocks' case.
Step 2: If blocks' case are in {1, 2 or 4}, apply formula (6) for extraction process
Step 3: If blocks' case are in {1, 2 or 3}, apply formula (7) for recovering process.
```

## 4. Experimental results

The Peak Signal-to-Noise Ratio (PSNR) and The Structural Similarity (SSIM) are two measurements that are usually used to assess the quality of two images. In our experiments, the PSNR is computed by following formula,

$$PSNR = 10 \times \log_{10}\left(\frac{255 \times 255}{MSE}\right) \quad (8)$$

MSE is Mean square error, which is calculated by,

$$MSE = \frac{1}{m \times n} \times \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\left(F(i,j) - N(i,j)\right) \quad (9)$$

Where m, n are row and column of images, F is original frame, and N is F's noisy approximation.

The SSIM index is used to measure the video quality. In this study, the SSIM index between the original frame and embedded frame is calculated by following formula,

$$SSIM = \frac{1}{N-1} \times \sum_{i=0}^{N-1} \frac{(2\mu_{O_i}\mu_{E_i} + c_1) \times (2\sigma_{O_iE_i} + c_2)}{(\mu^2{}_{O_i} + \mu^2{}_{E_i} + c_1) \times (\sigma^2{}_{O_i} + \sigma^2{}_{E_i} + c_2)} \quad (10)$$

where $O_i$ and $E_i$ denote $i$-th 4×4 luminance block in original frame and embedded frame; $N$ is number of 4×4 luminance blocks; $\mu_{O_i}, \mu_{E_i}$ and $\sigma^2{}_{O_i}, \sigma^2{}_{E_i}$ denote the mean variance of $O$ and $E$; $\sigma_{O_iE_i}$ is the covariance of $O$ and $E$; $c_1 = (k_1 \times L)^2$ and $c_2 = (k_2 \times L)^2$ with $L = 255$, $k_1 = 0.01$, and $k_2 = 0.03$.

In this study, The PSNR1 and SSIM1 are calculated to compare the embedded video with the original video, meanwhile, PSNR2 and SSIM2 are used compared to the decoded video of the H.264 files. Table 3 shows that the quality of videos when embedding maximum bits of videos for each quality parameters (QPs). The average of PSNR2 (35.94dB) is higher than average of PSNR1 (33.33dB), and the average of SSIM2 (0.952) is also higher than average of SSIM1 (0.848).

*Table 3. Quality of videos after embed for randomly secret data bits*

| | PSNR1 | SSIM1 | PSNR2 | SSIM2 |
|---|---|---|---|---|
| **22** | 38.21 | 0.942 | 40.18 | 0.977 |
| **24** | 36.24 | 0.915 | 38.36 | 0.970 |
| **26** | 34.23 | 0.880 | 36.75 | 0.961 |
| **28** | 32.48 | 0.842 | 35.20 | 0.951 |
| **30** | 30.38 | 0.784 | 33.54 | 0.937 |
| **32** | 28.41 | 0.722 | 31.59 | 0.918 |
| ***Avr*** | ***33.33*** | ***0.848*** | ***35.94*** | ***0.952*** |

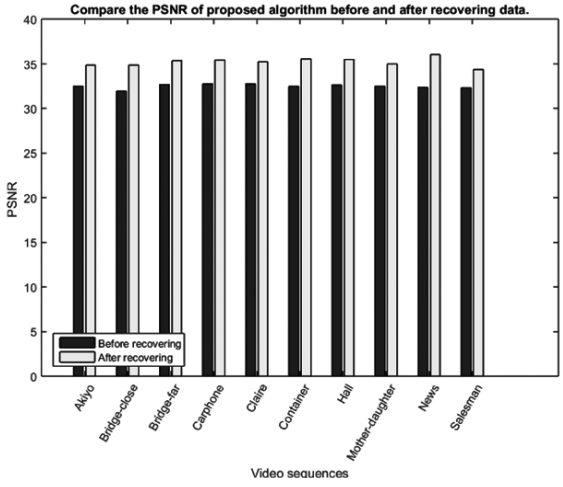


*Figure 6. Comparing the PSNR before and after recovering video with QP=28*

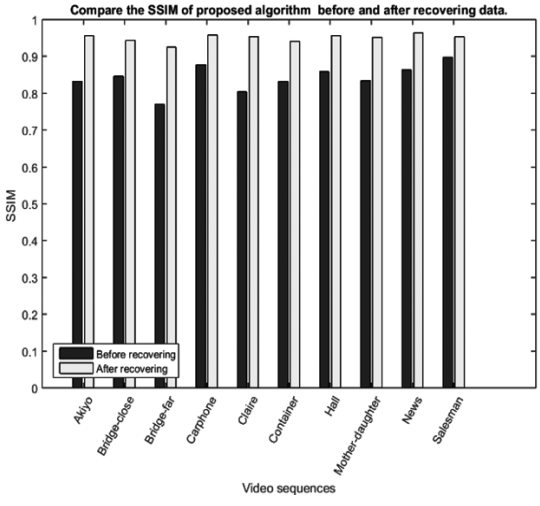


*Figure 7. Comparing the SSIM before and after recovering video with QP=28*

The deviation of PSNR and SSIM are about 2.61 and 0.104, respectively. For QP = 28, the max deviation of PSNR is 3.65dB with video sequence News (Figure 6). Meanwhile, the max deviation

of SSIM is 0.16 with video sequence Bridge-far (Figure 7).

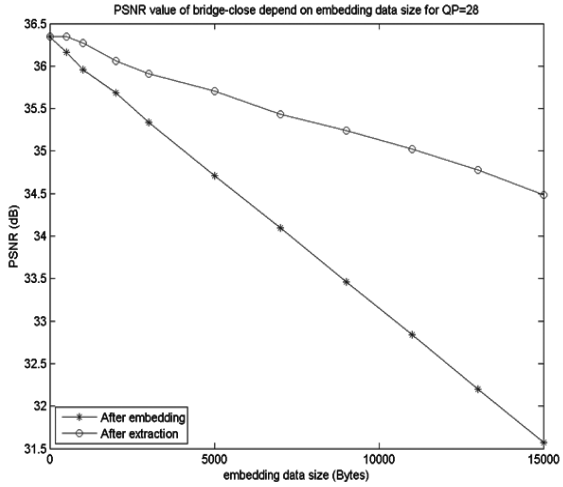


*Figure 8. PSNR of videos*

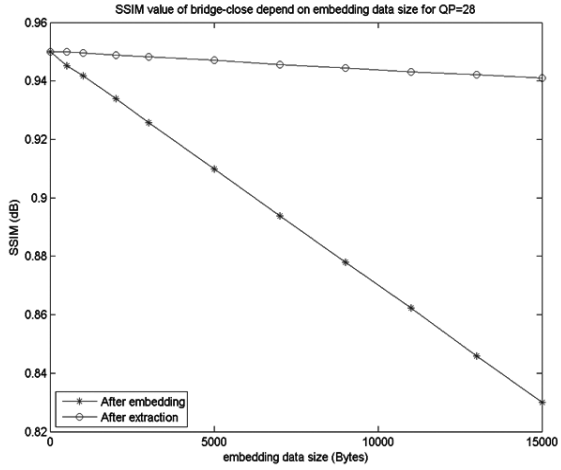


*Figure 9. SSIM of videos.*

By testing the quality of videos with difference embedding capacity (from 0 to 15000bits), we found that the higher deviation of PSNR and SSIM when embedding more capacity (Figure 8 and Figure 9).

In order to clearly know the effective of recovering videos, in this study, the authors embed two DNA sequences, which download from GenBank database. The study in [10] shows that the structure of embedding binary string is built from a DNA sequence, containing the sequence number, the size of DNA sequence, binary codes from nucleotides (nts). The DNA sequence consists four base type, is coded by A, G, C and T corresponding with 00, 01, 10 and 11, respectively. Each nucleotide is encrypted by 2 binary bits, so that, binary string size corresponding with DNA sequence NC_007020 is about 11440nts (~22880bits). For smaller DNA sequence size, NC_007203 (Table 4) with 6909nts (~13818bits), the deviation of average of PSNR1 and PSNR2 is 2.07dB, and the deviation of average of SSIM1 and SSIM2 is 0.079.

*Table 4. Quality of videos after embedding and recovering for DNA sequences NC_007203 with QP=28*

| | **PSNR1** | **SSIM1** | **PSNR2** | **SSIM2** |
|---|---|---|---|---|
| ***Akiyo*** | 34.58 | 0.878 | 36.10 | 0.959 |
| ***Bridge-close*** | 32.49 | 0.861 | 34.97 | 0.944 |
| ***Bridge-far*** | 34.78 | 0.831 | 36.49 | 0.929 |
| ***Carphone*** | 33.40 | 0.889 | 35.63 | 0.959 |
| ***Claire*** | 36.36 | 0.884 | 37.64 | 0.964 |
| ***Container*** | 33.30 | 0.854 | 35.90 | 0.942 |
| ***Hall*** | 33.70 | 0.882 | 36.08 | 0.959 |
| ***Mother-daughter*** | 34.38 | 0.876 | 36.14 | 0.955 |
| ***News*** | 33.66 | 0.889 | 36.41 | 0.966 |
| ***Salesman*** | 32.29 | 0.898 | 34.33 | 0.953 |
| ***Average*** | ***33.90*** | ***0.874*** | ***35.97*** | ***0.953*** |

Table 5 compares the PSNR, SSIM and maximum capacity of proposed algorithm with two algorithms, Ma et al. and Lin et al., for QP =28. Although, the PSNR of proposed algorithm (35.20dB) is lower the PSNR of Ma et al.'s algorithm (35.31dB), it seems higher when compare with Lin et al.'s algorithm (34.78). However, the SSIM and maximum capacity of proposed are always higher two algorithms [5, 6]. Especially, the proposed algorithm can reverse to the original video, while two algorithms cannot do.

*Table 5. Comparing the proposed algorithm with Ma et al.'s algorithm and Lin et al.'s algorithm for QP=28*

| | **Max capacity (bits)** | **PSNR (dB)** | **Reversibility** |
|---|---|---|---|
| ***Proposed algorithm*** | 26040 | 35.20 | Yes |
| ***Ma et al.'s algorithm*** | 11559 | 35.31 | No |
| ***Lin et al.'s algorithm*** | 14357 | 34.78 | No |

With the similar embedding capacity, the PSNR and SSIM of proposed algorithm are always higher with algorithms in [5] and [6], in term QP = 28.

## 5. Conclusion

Based on statistics of coefficient value and histogram shifting, an algorithm was performed, in order to embed as more capacity as possible, simultaneously, reduce distortion drift. The experimental results show that the proposed algorithm can embed more secret data into video and recover appropriately entire I-frames. With the development of genomics, the DNA sequences or other genomic sequences will be exchanged by two host, therefore, we can encrypt them to a binary string, and then embed into a video. This study just uses two DNA sequences for hiding into H.264/AVC video. For the future, a robustness of reversible data hiding in H.264/AVC, based on histogram shifting could be considered in order to suffer from the visual quality degradation.

## References

[1]. S. K. Kapotas, E. E. Varsaki, and A. N. Skodras, "Data Hiding in H. 264 Encoded Video Sequences," *IEEE 9th Workshop on Multimedia Signal Processing*, pp. 373-376, 2007.

[2]. T. S. Nguyen, C. C. Chang, M. C. Lin, "Adaptive lossless data-hiding and compression scheme for SMVQ indices using SOC," *Smart Comput. Review,* **vol. 4, no. 3**, pp. 230-245, 2014.

[3]. C. C. Chang, T. S. Nguyen, "A reversible data hiding scheme for SMVQ indices," *Informatica,* **vol. 25, no. 4**, pp. 523-540, 2014.

[4]. E. G. Richardson, "H.264 and MPEG-4 video compression: video coding for next-generation multimedia," *Chichester, U.K.: Wiley*, 2003.

[5]. X. J. Ma, Z. T. Li, H. Tu, B. Zhang, "A data hiding algorithm for H.264/AVC video streams without intra-frame distortion drift," *IEEE Trans. Circuits Systs. Video Technol.*, **vol. 20,** pp. 1320–1330, 2010.

[6]. T. J. Lin, K. L. Chung, P. C. Chang, Y. H. Huang, H. Y. M. Liao, C. Y. Fang, "An improved DCT-based perturbation scheme for high capacity data hiding in H.264/AVC intra frames," *The Journal of Systems and Software*, **vol. 86**, pp. 604-614, 2013.

[7]. Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Trans. Circuits Syst. Video Technol*., **vol. 16, no. 3**, pp. 354–362, Mar. 2006.

[8]. J.D. Watson, F.H.C. Crick, "Molecular structure of Nucleic acids: A structure for deoxyribose nucleic acid," *Nature* **171**, pp. 737, 738, 1953.

[9]. A. Muhit, M. R. Pickering, M. R. Frater and J. F. Arnold, "Video Coding Using Elastic Motion Model and Larger Blocks," *IEEE Transactions on Circuits and Systems for Video Technology*, **vol. 20, no. 5**, pp. 661-672, May 2010.

[10]. Chien. N. D, Son N. T, & Hsu F. R, "An algorithm for DNA sequence hiding in H. 264/AVC video." *Proceedings of the Seventh Symposium on Information and Communication Technology ACM*, pp. 229-234, December 2016.

## THUẬT TOÁN GIẤU TIN CÓ KHẢ NĂNG PHỤC HỒI TRONG VIDEO H.264/AVC

**Tóm tắt:**

*Giấu tin có khả năng phục hồi là một kỹ thuật nhúng dữ liệu mật vào một vật chủ như ảnh, cơ sở dữ liệu, âm thanh, video, song nó có thể phục hồi vật chủ gốc. Với kỹ thuật dịch chuyển biểu đồ, trong bài báo này, một thuật toán giấu tin có khả năng phục hồi trong video H.264/AVC được đề xuất với mục đích tăng khả năng nhúng lớn nhất có thể, đồng thời, có thể phục hồi video gốc tốt nhất. Nghiên cứu này cũng có thể tránh được sự biến dạng của video. Các kết quả thí nghiệm cho thấy thuật toán đề xuất có thể phục hồi xấp xỉ video gốc. Việc so sánh các nghiên cứu khác, nghiên cứu đề xuất cải thiện hơn nữa khả năng nhúng, và có thể phục hồi video gốc. Điểm yếu của thuật toán này là không thể khôi phục lỗi bit khi xảy ra tấn công mạng. Trong tương lai, chúng tôi sẽ sử dụng kỹ thuật mã hóa BCH để tang tính mạnh mẽ của việc giấu tin cho thuật toán đề xuất.*

***Từ khóa:*** *Giấu tin có khả năng phục hồi, DCT, H.264/AVC, khả năng nhúng, biến dạng, dịch chuyển biểu đồ.*