International Journal of Engineering Research and Reviews ISSN 2348-697X (Online)
Vol. 3, Issue 2, pp: (71-80), Month: April - June 2015, Available at: www.researchpublish.com

Data Hiding in Encrypted H.264/AVC Video Streams

[1]Arun Kumar Mogaparthi, [2]Thirupathi Alle, [3]Varun Anand, [4]Akash Ramteke

Abstract: **The upcoming H.264/MPEG-4 AVC video compression standard promises a significant improvement over all previous video compression standards. In terms of coding efficiency, the new standard is expected to provide at least 2x compression improvement over the best previous standards and substantial perceptual quality improvements over both MPEG-2 and MPEG-4. Video encryption often requires that the scheme be time efficient to meet the requirement of real time and format compliance. It is not practical to encrypt the whole compressed video bitstream like what the traditional ciphers do because of the following two constraints, i.e., format compliance and computational cost. Alternatively, only a fraction of video data is encrypted to improve the efficiency while still achieving adequate security. The key issue is then how to select the sensitive data to encrypt. Iit is reasonable to encrypt both spatial information (IPMand residual data) and motion information (MVD) during H.264/AVC encoding.**

Keywords: **Data Hiding in Encrypted, H.264/AVC Video Streams, MPEG-2 and MPEG-4.**

1. INTRODUCTION

We all know that cloud computing has become an important technology, which can provide highly efficient computation and large scale storage solution for video data. But it is been proven that cloud services are more vulnerable to attacks. So, a better solution for this is encrypt videos and then store them on cloud. the capability of hiding data in encrypted H.264/AVC video streams would avoid video content leakage. this will help in addressing the security and privacy concerns. Different techniques can be used for data hiding and encrypting videos.

Data hiding is the process of embedding information into a host medium. In general, visual and aural media are preferred due to their wide presence and the tolerance of human perceptual systems involved. Although the general structure of data hiding process does not depend on the host media type, the methods vary depending on the nature of such media. For instance, image and video data hiding share many common points; however video data hiding necessitates more complex designs, as a result of the additional temporal dimension. Therefore, video data hiding continues to constitute an active research area.

Data hiding in video sequences is performed in two major ways: bit stream-level and data-level. In bit stream-level, the redundancies within the current copression standards are exploited. Typically, encoders have various options during encoding and this freedom of selection is suitable for manipulation with the aim of data hiding.

However, these methods highly rely on the structure of the bit stream; hence, they are quite fragile, in the sense that in many cases they cannot survive any format conversion or transcoding, even without any significant loss of perceptual quality.

As a result, this type of data hiding methods is generally proposed for fragile applications, such as authentication. On the other hand, data-level methods are more robust to attacks. Therefore, they are suitable for a broader range of applications. Despite their fragility, the bit stream-based methods are still attractive for data hiding applications.

For instance, in, the redundancy in block size selection of H.264 encoding is exploited for hiding data. In another approach, the quantization parameter and DCT (Discrete Cosine Transform) coefficients are altered in the bitstream-level. However, most of the video data hiding methods utilize uncompressed video data. They apply QIM to low-frequency DCT

coefficients and adapt the quantization parameter based on MPEG-2 parameters. Furthermore, they vary the embedding rate depending on the type of the frame. As a result, insertions and erasures occur at the decoder, which causes desynchronization. They utilize Repeat Accumulate (RA) codes in order to withstand erasures. Since they adapt the parameters according to type of frame, each frame is processed separately RA codes are already applied in image data hiding.

Therefore, it becomes desirable to develop data hiding algorithms that work entirely on encode bitstream in the encrypted domain. However, there are challenges for hiding data in directly encrypted bitstream. Based on the analysis given above, we propose a novel scheme to embed data directly in encrypted H.264/AVC video bitstream.

The proposed scheme can achieve high performance in the following three prospects:- The data hiding s performed in directly encrypted H.264/AVC video bitstream, Format compliance and strict file size preservation & The scheme can be applied for two different application scenarios by extracting the hidden data either from encrypted video stream or from decrypted video. The algorithm we are using for encrypting H.264/AVC compressed video is RC4. RC4 (Ron's Code) is a stream cipher and is used in popular protocols such as Transport layer security(TLS) and WEP. It is more efficient without decryption followed by data hiding and re-encryption.

2. PROPOSED SYSTEM

The proposed can be divided into two modules/sections

1. Encryption Section
2. Decryption Section

Each Section is further divided into sub-sections so as to have a proper structure and for flexibility.

1. Encryption Section:

In this section the H.264/AVC video is first compressed using the H.264 codec, then encrypted by an encryption algorithm, here we are using RC4 algorithm for this purpose. Finally the data is embedded in the encrypted video stream. Fig 1 show the pictorial representation of this section

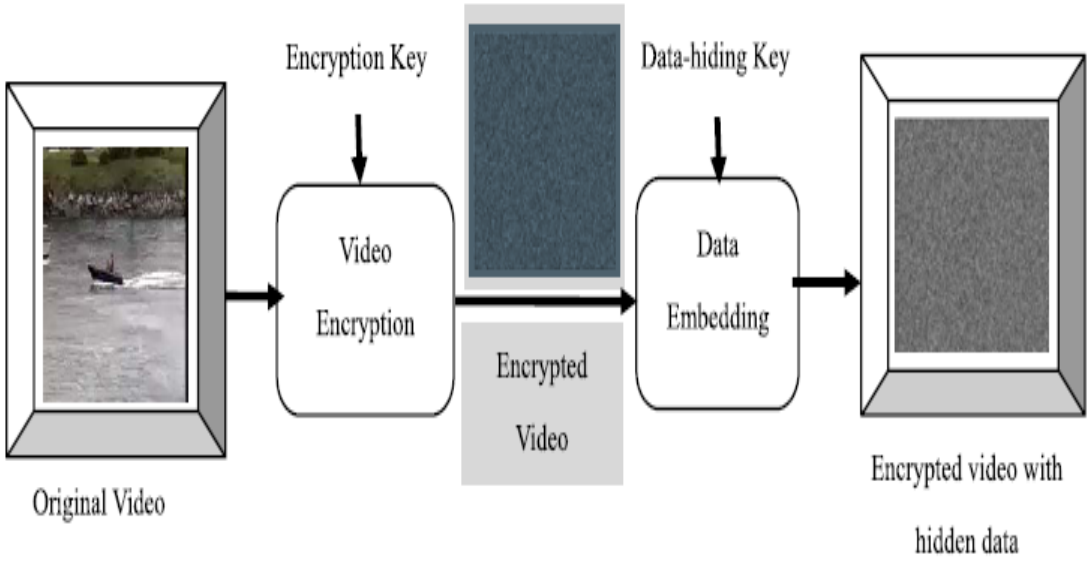

Fig 1. Video encryption and data embedding at the sender end.

A. Video compression using H.264:

By analyzing the property of H.264/AVC codec, the codewords of intraprediction modes, the codewords of motion vector differences, and the codewords of residual coefficients are encrypted with stream ciphers. Then, a data hider may embed additional data in the encrypted domain by using codeword substitution technique, without knowing the original video content. In order to adapt to different application scenarios, data extraction can be done either in the encrypted domain or in the decrypted domain. Furthermore, video file size is strictly preserved even after encryption and data embedding.

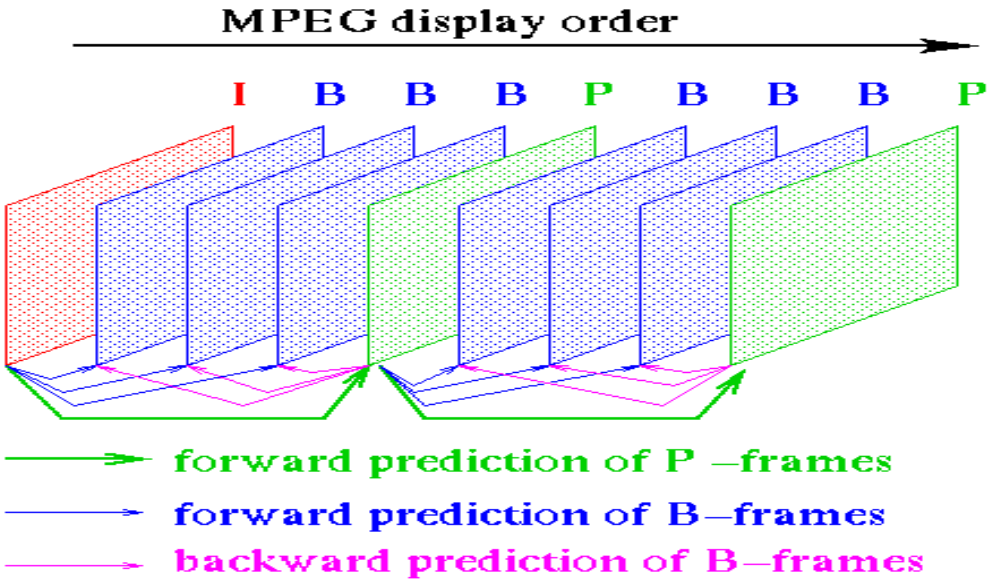

Fig: 2. MPEG display order

In Fig. 2, The I-frames are *intra coded*, i.e. they can be reconstructed without any reference to other frames. The P-frames are forward predicted from the last I-frame or P-frame, i.e. it is impossible to reconstruct them without the data of another frame (I or P). The B-frames are both, forward predicted and backward predicted from the last/next I-frame or P-frame, i.e. there are two other frames necessary to reconstruct them. P-frames and B-frames are referred to as *inter coded* frames.

The upcoming H.264/MPEG-4 AVC video compression standard promises a significant improvement over all previous video compression standards. In terms of coding efficiency, the new standard is expected to provide at least 2x compression improvement over the best previous standards and substantial perceptual quality improvements over both MPEG-2 and MPEG-4.

Video compression is about reducing and removing redundant video data so that a digital video file can be effectively sent and stored. The process involves applying an algorithm to the source video to create a compressed file that is ready for transmission or storage. To play the compressed file, an inverse algorithm is applied to produce a video that shows virtually the same content as the original source video.

The time it takes to compress, send, decompress and display a file is called latency. The more advanced the compression algorithm, the higher the latency, given the same processing power. A pair of algorithms that works together is called a video codec (encoder/decoder). Video codecs that implement different standards are normally not compatible with each other; that is, video content that is compressed using one standard cannot be decompressed with a different standard.

For instance, an MPEG-4 Part 2 decoder will not work with an H.264 encoder. This is simply because one algorithm cannot correctly decode the output from another algorithm but it is possible to implement many different algorithms in the same software or hardware, which would then enable multiple formats to be compressed. Different video compression standards utilize different methods of reducing data, and hence, results differ in bit rate, quality and latency. Results from encoders that use the same compression standard may also vary because the designer of an encoder can choose to implement different sets of tools defined by a standard.

As long as the output of an encoder conforms to a standard's format and decoder, it is possible to make different implementations. This is advantageous because different implementations have different goals and budget. Professional non-real-time software encoders for mastering optical media should have the option of being able to deliver better encoded video than a real-time hardware encoder for video conferencing that is integrated in a hand-held device.

A given standard, therefore, cannot guarantee a given bit rate or quality. Furthermore, the performance of a standard cannot be properly compared with other stan dards, or even other implementations of the same standard, without first defining how it is implemented. A decoder, unlike an encoder, must implement all the required parts of a standard in order to decode a compliant bit stream. This is because a standard specifies exactly how a decompression algorithm should restore every bit of a compressed video.

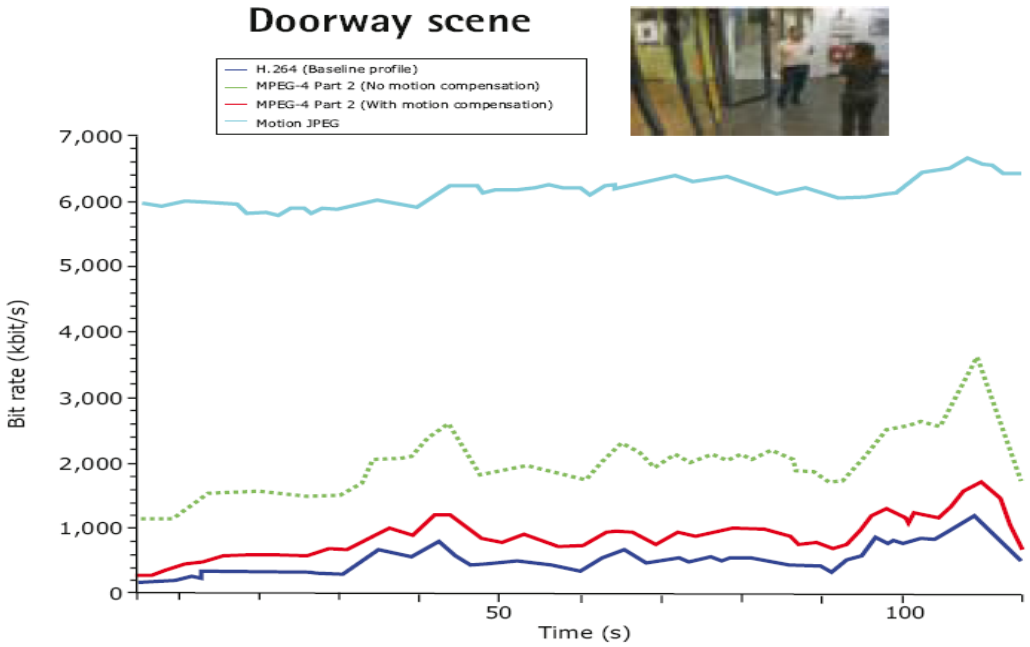

Fig: 3. Comparision of H.264/AVC Codec with others

```
S - State vector
T - Temporary Vector
k - key
P - Plaintext

Initial permutation on S

    j=0;
    for i=0 to 7 do
        j=(j+s[i] + t[i]) % stream_length
        swap(s[i], s[j])
    end for
```

Fig 4: RC4 Pseudocode

B. Video encryption using RC4:

RC4 generates a pseudorandom stream of bits (a keystream). As with any stream cipher, these can be used for encryption by combining it with the plaintext using bit-wise exclusive-or; decryption is performed the same way, except that generated *pseudorandom bits*, rather than a prepared stream, are used.

To generate the keystream, the cipher makes use of a secret internal state which consists of two parts:

1. A permutation of all 256 possible bytes
2. Two 8-bit index-pointers (denoted "i" and "j").

The permutation is initialized with a variable length key, typically between 40 and 256 bits, using the key-scheduling algorithm (KSA). Once this has been completed, the stream of bits is generated using the *pseudo-random generation algorithm* (PRGA).

RC4, being a stream cipher, was for a period of time the only common cipher that was immune to the 2011 BEAST attack on TLS 1.0. The attack exploits a known weakness in the way cipher block chaining mode is used with all of the other ciphers supported by TLS 1.0, which are all block ciphers. In March 2013, there were new attack scenarios proposed by Isobe, Ohigashi, Watanabe and Morii,[25] as well as AlFardan, Bernstein, Paterson, Poettering and Schuldt that use new statistical biases in RC4 key table to recover plaintext with large number of TLS encryptions.

C. Data Embedding in Video Stream:

Even Though, quite enough strategies are planned to embed the data directly into H.264/AVC bitstream, however, these strategies cannot be enforced within the encrypted domain.In the encrypted bitstream of H.264/AVC, the proposed data embedding is accomplished by replacing eligible codewords of Levels in Table I. Since the sign of Levels are encrypted, data hiding must not have an effect on the sign of Levels.

Besides, the codewords substitution ought to satisfy the subsequent 3 limitations.

First, the bitstream once codeword work should stay syntax compliance so it will be decoded by normal decoder. Second, to stay the bit-rate unchanged, the substituted codeword ought to have a similar size because the original codeword. Third, information concealing will cause visual degradation however the impact ought to be unbroken to minimum. That is, the embedded information once video coding must be invisible to a person's observer.

So the price of Level love the substituted co-deword ought to keep near the value of Level love the first codeword. Additionally, the codewords of Levels among P-frames square measure used for datahiding, whereas the codewords of Levels in I-frames square measure remained unchanged. as a result of I-frame is that the initial frame a gaggle of images (GOPs), the error occurred in I-frame are going to be propagated to succeeding P-frames.

According to the analysis given on top of, we will see that there aren't any corresponding substituted codewords once suffixLength is adequate to zero or one, as shown in Table III. once suffixLength is adequate to zero, we have a tendency to cannot realize a try of codewords with a similar size.

When suffixLength is adequate to one, one codeword additionally can not be substituted by another codeword with a similar size, since this substitution would modification the sign of Level. Then the co-dewords of Levels that suffixLength is a pair of or three would bedivided into 2 opposite codespaces denoted as C0and C1.

The code-words appointed in C0and C1 square measure related to binary hidden data “0” and “1”.

Suppose the extra information that we would like to enter may be a bi-nary sequence denoted as,

$$B = \{b(i)|i = one, 2, \bullet\bullet\bullet, L, b(i) \in \}.$$

Data concealing is performed directly in encrypted bitstream through the subsequent steps.

Step 1. so as to boost the safety, the extra information is encrypted with the chaotic pseudo-random sequence,

$$P = \{p(i)|i = one, 2, \bullet\bullet\bullet, L, p(i) \in \}$$

to generate the to-be-embedded sequence,

$$W = \{w(i)|i = one, 2, \bullet\bullet\bullet, L,w(i) \in \}.$$

The sequence P is generated by victimization logistical map with associate degree initial worth, i.e., the information concealing key. it's terribly troublesome for anyone UN agency doesn't retain {the information encryption key to recover the extra data.

Step 2. The codewords of Levels square measure obtained by parsing the Encrypted H.264/AVC bitstream.

Step 3. If current codeword belongs to codespaces C0 or C1, the to-be-embedded information bit will be embedded by codeword substi-tuting. Otherwise, the codeword is left unchanged. The elaborate procedure of codeword work for information concealing. For exam-ple, once Level is positive one and its sufflxLength is three, then its corresponding codeword is “1000” that belongs to C0 as shown in Fig. 2(c). If the information bit “1” has to be embedded, the codeword “1000” ought to get replaced with “1010”. Otherwise, if the information bit “0” has to embedded, the codeword “1000” can keep unchanged.

Step 4. Select consequent codeword and so move to Step3 for data hiding. If there aren't any a lot of information bits to be embedded, the embedding method is stopped. Suppose the to-be-embedded information is “1001”, the CAVLC codeword of Level parsing from H.264/AVC bitstream is “0101000100001000001010110000100” and also the secret writing stream is “10111”, associate degree example of information embedding supported codeword mapping.

D. Data Extraction:

Here, the hidden data is extracted either in encrypted or decrypted domain, as shown in Fig. 5.data extraction method is quick and straightforward. we'll 1st discuss the ex-traction in encrypted domain followed by decrypted domain.

Table 1: Codeword Substiution Table

Sr. No.	Level	Codeword	Substitution
1	1	100	110
2	2	0100	0110
3	3	00100	00110
4	4	000100	000110
5	5	0000100	0000110
6	6	00000100	00000110
7	7	000000100	000000110
8	1	101	111
9	2	0101	0111
10	3	00101	00111
11	4	000101	000111
12	5	0000101	0000111
13	6	00000101	00000111
14	7	000000101	000000111
15	1	1000	1010
16	2	1100	1110
17	3	01000	01010
18	4	01100	01110
19	5	001000	001010
20	6	001100	001110
21	7	0001000	0001010
22	1	1001	1011
23	2	1101	1111
24	3	01001	01011
25	4	01101	01111
26	5	001001	001011
27	6	001101	001111
28	7	0001001	0001011

Encrypted Domain Extraction. to guard privacy, a database manager (e.g., cloud server) might solely get access to the hidden information, encryption key and need to manipulate data in encrypted domain. Data extraction in encrypted do-

main guarantees the practicability of our theme during this case. In encrypted domain, as shown in 5, encrypted video with hidden knowledge is directly sent to the info extraction module, and also the extraction method is given as follows.

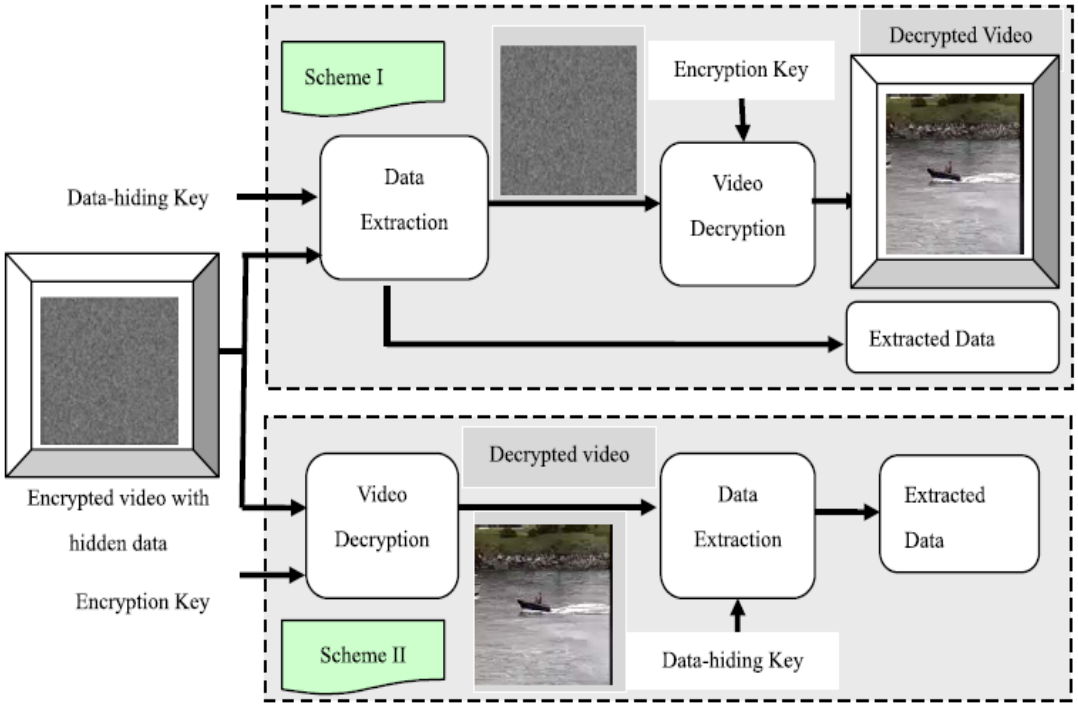

Fig 5: Data Extraction & Decryption Model

Step1: The codewords of Levels area unit first known by parsing the encrypted bitstream.

Step2: If the codeword belongs to codespace C0, the extracted knowledge bit is "0". If the codeword belongs to codespace C1, the extracted knowledge bit is "1".

Step3: In keeping with the data encryption key, an equivalent chaotic pseudo – random sequence P that was utilized in the embedding method is generated.

Then the extracted bit sequence can be decrypted by victimization P to urge the first extra info. Since the total method is entirely operated in encrypted do-main, it effectively avoids the outflow of original video content. associate example of knowledge extraction in encrypted domain

3. EXPERIMENT AND RESULTS

The proposed data hiding scheme has been implemented in the H.264/AVC reference software version JM-12.2. Sixwell-known standard video sequences (i.e., *Stefan*, *Table*, *Tempete*, *Mobile*, *Hall*, *and News*) in QCIF format (176 × 144) at the frame rate 30 frames/s are used for simulation. The first 100 frames in each video sequence are used in the experiments. The GOP

(Group of Pictures) structure is "IPPPP: one I frame followed four P frames".

For the proposed video encryption scheme, the security includes both cryptographic security and perceptual security. Cryptographic security denotes the security against cryptographic attacks, which depends on the ciphers adopted by the scheme. In the proposed scheme, the secure stream cipher (e.g., RC4) is used to encrypt the bitstream, and chaotic pseudo-random sequence generated by logistic map is used to encrypt the additional data. They have been proved to be secure against cryptographic attacks. Perceptual security refers to whether the encrypted video is unintelligible or not.

Generally, it depends on the encryption scheme's properties. For example, encrypting only IPM cannot keep secure enough, since the encrypted video is intelligible. The proposed scheme encrypts IPM, MVD and residual coefficients, which keeps perceptual security of the encrypted video. The demonstration is shown in Figs. 6 and 7. An original frame from each video is depicted in Fig. 6, and their corresponding encrypted results are depicted in Fig. 7. Other frames have a

similar effect of encryption. Due to space limitations, we do not list the results of all frames. It should be mentioned that not every video can be degraded to the same extent. The perceptual quality of high-motion videos with a complex textured background becomes much more scrambled after encryption than that of low-motion videos with a static background. The reason is that there are less residual coefficients and MVDs in videos that are available for encryption. In general, scrambling performance of the described encryption system is more than adequate.

The encrypted video containing hidden data provided by the server should be decrypted by the authorized user. Therefore, the visual quality of the decrypted video containing hidden data is expected to be equivalent or very close to that of the original video. By modifying the compressed bitstream to embed additional data, the most important challenge is to maintain perceptual transparency, which refers to the modification of bitstream should not degrade the perceived content quality. In this paper, only the codewords of Levels within P-frames are modified for data hiding. Simulation results have demonstrated that we can embed the additional data with a large capacity into P-frames while preserving high visual quality. The encrypted and decrypted video frames with hidden data are shown in Figs 8 and 9 respectively. In the experiments, no visible artifacts have been observed in all of the decrypted video frames with hidden data.

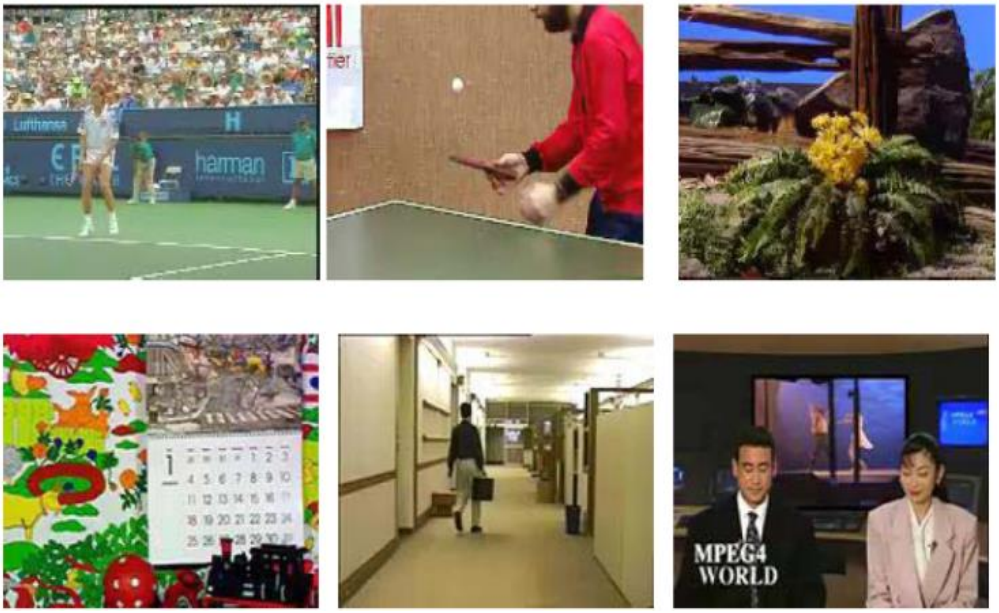

Fig. 6: Original Video Frames

Fig 7.Encrypted Video Frames

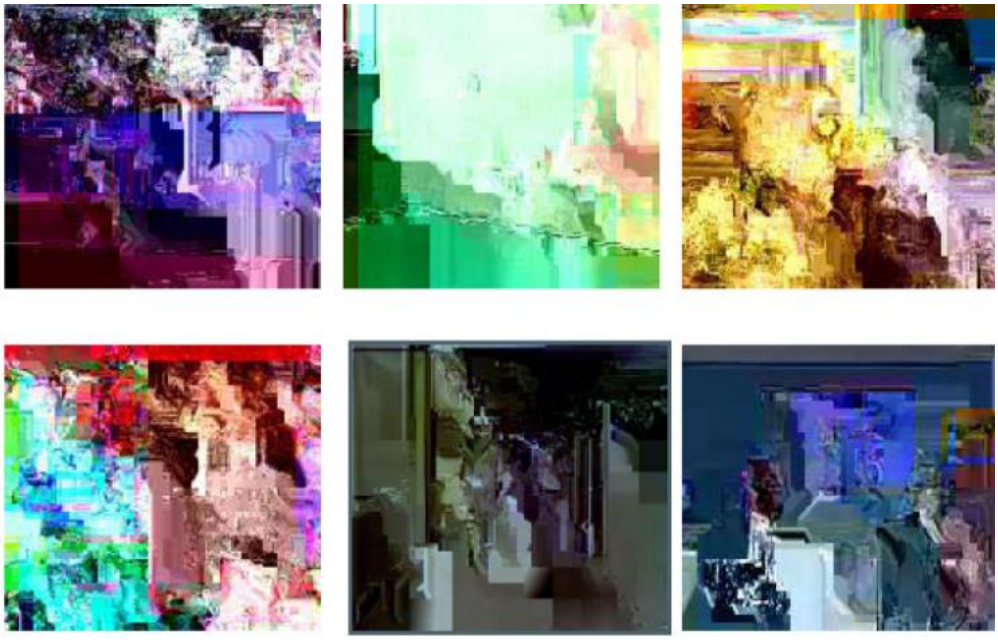

Fig. 8 Encrypted Video frames with hidden data

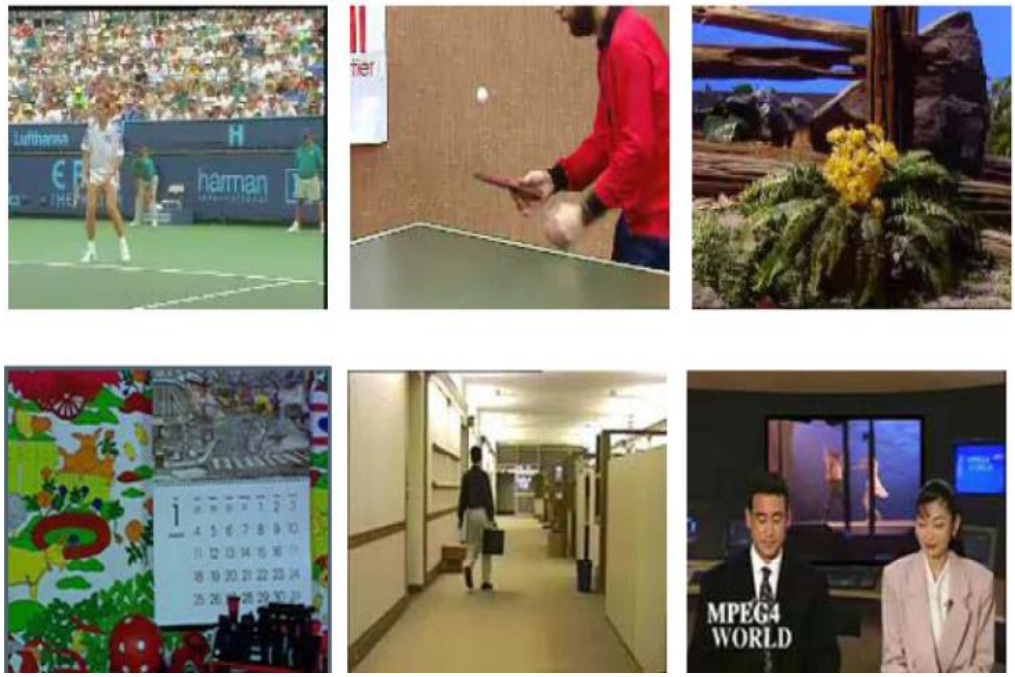

Fig. 9. Dcrypted video frames with hidden data

4. CONCLUSION

To the best of our knowledge, till now, there is no algorithm to embed additional data directly in encrypted H.264/AVC video stream. Therefore, no detailed experimental comparisons are given in the paper. As described in Section I, in the existing related technologies, encryption and data hiding are accomplished almost simultaneously during H.264/AVC encoding process. In addition, encryption and data embedding would lead to increasing the bit-rate of H.264/AVC bitstream. On the contrary, our proposed scheme can encrypt H.264/AVC video stream directly and then embeds data into encrypted H.264/AVC video stream to meet the privacy-preserving requirements. The bit-rate of the encrypted H.264/AVC video stream containing hidden data is exactly the same as the original H.264/AVC video stream.

Data hiding in encrypted media is a new topic that has started to draw attention because of the privacy-preserving requirements from cloud data management. In this paper, an algorithm to embed additional data in encrypted H.264/AVC

bitstream is presented, which consists of video encryption, data embedding and data extraction phases. The algorithm can preserve the bit rate exactly even after encryption and data embedding, and is simple to implement as it is directly performed in the compressed and encrypted domain, i.e., it does not require decrypting or partial decompression of the video stream thus making it ideal for real-time video applications. The data-hider can embed additional data into the encrypted bitstream using codeword substituting, even though he does not know the original video content. Since data hiding is completed entirely in the encrypted domain, our method can preserve the confidentiality of the content completely. With an encrypted video containing hidden data, data extraction can be carried out either in encrypted or decrypted domain, which provides two different practical applications. Another advantage is that it is fully compliant with the H.264/AVC syntax. Experimental results have shown that the proposed encryption and data embedding scheme can preserve file-size, whereas the degradation in video quality caused by data hiding is quite small.

REFERENCES

[1] Dawen Xu, Rangding Wang, and Yun Q. Shi, Fellow," Data Hiding in Encrypted H.264/AVC Video Streams by Codeword Substitution", IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, VOL. 9, NO. 4, APRIL 2014.

[2] W. J. Lu, A. Varna, and M. Wu, "Secure video processing: Problems and challenges," in Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, Prague, Czech Republic, May 2011, pp. 5856–5859.

[3] B. Zhao, W. D. Kou, and H. Li, "Effective watermarking scheme in the encrypted domain for buyer-seller watermarking protocol," Inf. Sci., vol. 180, no. 23, pp. 4672–4684, 2010.

[4] P. J. Zheng and J. W. Huang, "Walsh-Hadamard transform in the homomorphic encrypted domain and its application in image watermarking," in Proc. 14th Inf. Hiding Conf., Berkeley, CA, USA, 2012, pp. 1–15.

[5] W. Puech, M. Chaumont, and O. Strauss, "A reversible data hiding method for encrypted images," Proc. SPIE, vol. 819, pp. 68191E-1–68191E-9, Jan. 2008.

[6] X. P. Zhang, "Reversible data hiding in encrypted image," IEEE Signal Process. Lett., vol. 18, no. 4, pp. 255–258, Apr. 2011.

[7] W. Hong, T. S. Chen, and H. Y. Wu, "An improved reversible data hiding in encrypted images using side match," IEEE Signal Process. Lett., vol. 19, no. 4, pp. 199–202, Apr. 2012.

[8] X. P. Zhang, "Separable reversible data hiding in encrypted image," IEEE Trans. Inf. Forensics Security, vol. 7, no. 2, pp. 826–832, Apr. 2012.

[9] K. D. Ma, W. M. Zhang, X. F. Zhao, N. Yu, and F. Li, "Reversible data hiding in encrypted images by reserving room before encryption," IEEE Trans. Inf. Forensics Security, vol. 8, no. 3, pp. 553–562, Mar. 2013.

[10] A. V. Subramanyam, S. Emmanuel, and M. S. Kankanhalli, "Robust watermarking of compressed and encrypted JPEG2000 images," IEEE Trans. Multimedia, vol. 14, no. 3, pp. 703–716, Jun. 2012.

[11] S. G. Lian, Z. X. Liu, and Z. Ren, "Commutative encryption and watermarking in video compression," IEEE Trans. Circuits Syst. Video Technol., vol. 17, no. 6, pp. 774–778, Jun. 2007.

[12] S. W. Park and S. U. Shin, "Combined scheme of encryption and watermarking in H.264/scalable video coding (SVC)," New Directions Intell. Interact. Multimedia, vol. 142, no. 1, pp. 351–361, 2008.

[13] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 7, pp. 560–576, Jul. 2003.

[14] S. G. Lian, Z. X. Liu, Z. Ren, and H. L. Wang, "Secure advanced video coding based on selective encryption algorithms," IEEE Trans. Consumer Electron., vol. 52, no. 2, pp. 621–629, May 2006.

[15] Z. Shahid, M. Chaumont, and W. Puech, "Fast protection of H.264/AVC by selective encryption of CAVLC and CAAC for I and P frames, "IEEE Trans. Circuits Syst. Video Technol., vol. 21, no. 5, pp. 565–576, May 2011.

[16] M. N. Asghar and M. Ghanbari, "An efficient security system for CABAC bin-strings of H.264/SVC," IEEE Trans. Circuits Syst. Video Technol., vol. 23, no. 3, pp. 425–437, Mar. 2013.