Cube-based Encryption Connected Prior to Motion JPEG Standard

Kosuke Shimizu* and Taizo Suzuki†
* Department of Computer Science, University of Tsukuba, Japan
E-mail: shimizu@adapt.cs.tsukuba.ac.jp
† Faculty of Engineering, Information and Systems, University of Tsukuba, Japan
E-mail: taizo@cs.tsukuba.ac.jp

***Abstract*—We propose a cube-based encryption-then-compression (ETC) system for the Motion JPEG standard. ETC systems reduce the visibility of multimedia content, such as a video sequence, to provide a high level of security at the same time as compressing it efficiently. The conventional ETC system for Motion JPEG applies block-based encryption to each frame of a sequence. However, although the quality of the encrypted and compressed frames is comparable to that of the unencrypted frames, any third party may decrypt the content from a frame without a decryption key because the blocks cannot be moved from one frame to another. To enhance the security of the ETC system, we propose a cube-based encryption by extending the block-based encryption. Experiments show that the proposed encryption method affects the Motion JPEG in quality, whereas that the level of encryption is high both objectively and subjectively.**

I. Introduction

Telecommunications of multimedia content on bandwidth-constrained and unsecured channels of, e.g., cloud computing or social networking services (SNSs), face two problems: the sender **Alice** may have difficulty sending large amounts of content to a receiver **Bob**, and a third party may be able to look at and/or modify the sent content illegally. To send content securely and efficiently, the content should be encrypted before or after compression. Here, the compression-then-encryption (CTE) system (Fig. 1(a)) encrypts the content after it has been compressed [1]–[4]. However, since the file format of the encrypted content is restricted to `.aes` (the Advanced Encryption Standard (AES) [5]), the encrypted content cannot be compressed efficiently and is inapplicable to services that cannot handle AES files.

The encryption-then-compression (ETC) system, which encrypts the content before it is compressed (Fig. 1(b)), has been suggested as a better means of privacy protection compared with CTE system [6]–[9]. The ETC system usually uses a "perceptual" encryption, which encrypts the content visually while keeping its file format unaltered. Therefore, content encrypted by ETC system can be compressed with high compression technology such as the JPEG standard and is applicable to many services offered today. Kurihara et al. proposed a block-based encryption for the Motion JPEG standard (Fig. 2(a)). A bitstream, consisting of content encrypted and compressed by **Alice**, is sent to **Bob**. **Bob** obtains the processed content by decompressing and decrypting the bitstream. The system can compress each encrypted frame with the same compression ratio as the unencrypted one. However, a frame may be illegally decrypted from each frame without any encryption key; i.e., the security of ETC system is insufficient.

To enhance the security of ETC system, we propose a cube-based encryption for Motion JPEG. The system, shown in Fig. 2(b), encrypts each frame not only spatially but also temporally and guards against illegal decryption from only one frame. To this end, we consider a video sequence to be a large cuboid and divide it up into smaller cubes that do not overlap. Since several blocks in the each encrypted frame have spatiotemporal sections obtained by arranging the frames temporally, illegal decryption from a single frame is clearly impossible. Experiments show that the proposed encryption method affects the Motion JPEG in quality, whereas the level of encryption is high both objectively and subjectively.

II. Block-based ETC System for Motion JPEG

A. Block-based Encryption

A block-based encryption [9] is shown in Fig. 2(a). The block scrambling randomly permutes the blocks, into which a frame has been divided such that they do not overlap. The block rotation and inversion randomly rotates and inverts the blocks. The directions of block rotation and block inversion are $\{0°, 90°, 180°, 270°\}$ and {non inversion, horizontal inversion, vertical inversion, horizontal+vertical inversion}, respectively. The block negative/positive transformation randomly inverts each color of the blocks. Let C_i, C'_i, and rand(i) be either of $\{R_i, G_i, B_i\}$, which are the pixel values of ith block, the encrypted values based on C_i, i.e.,

$$C'_i = \begin{cases} 255 - C_i & (\text{rand}(i) = 0) \\ C_i & (\text{otherwise}) \end{cases},$$

and a binary random value calculated at the block. The block color component shuffling rearranges the order of the color components in each block randomly; e.g., the pixel values $\{R_i, G_i, B_i\}$ may be rearranged into $\{G_i, B_i, R_i\}$.

B. Weak Point of Block Rotation/Inversion

A weak point of block rotation/inversion is that all blocks must remain in the same frame. This means that a third party may be able to decrypt the all encrypted blocks in the same

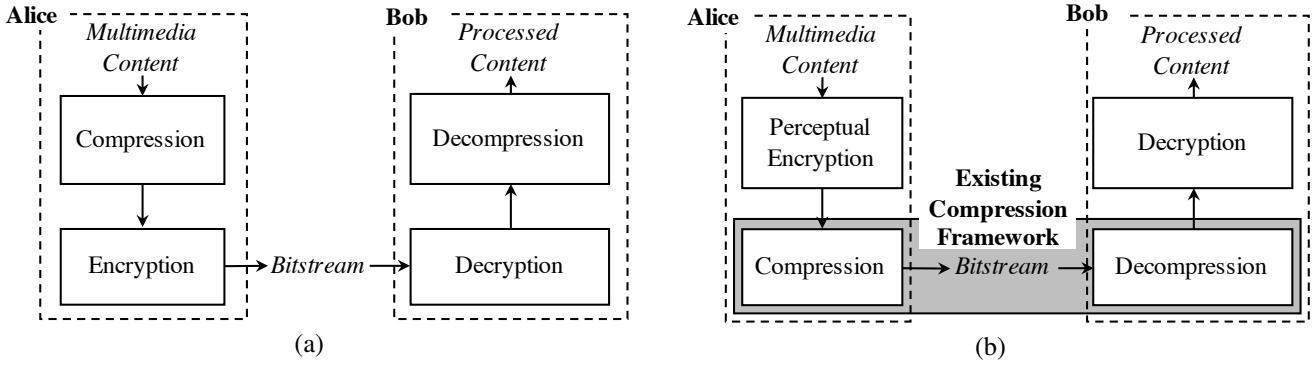

Fig. 1. Procedure of encryption and compression: (a) CTE system, (b) ETC system.

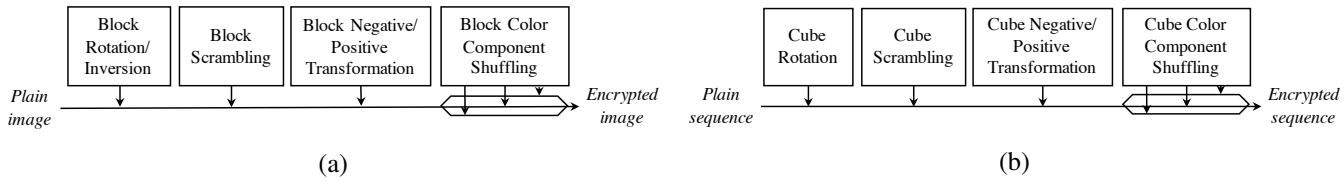

Fig. 2. Block/cube-based encryption: (a) block case, (b) cube case.

frame illegally by using a jigsaw puzzle solver [10] and a brute-force attack, even if it does not have the decryption key or infinite computational power. To solve this problem, at least one encrypted block must not remain in the original frame.

III. Cube-based ETC System for Motion JPEG

A. Cube-based Encryption

The cube-based ETC system consists of four encryption methods: which are called cube rotation, cube scrambling, cube negative/positive transformation, and cube color component shuffling. Regarding the new methods, we consider a video sequence to be a large cuboid and divide it into smaller cubes that do not to overlap each other. The sequence is encrypted by the new methods.

In the cube rotation shown in Fig. 3, each cube is rotated in a random direction and through a random angle. The directions and angles are {vertically, horizontally, depth-wise} and $\{0°, 90°, 180°, 270°\}$, respectively. The details are described in Algorithm 1. B_v, B_h, and B_d are the height, width, and depth of a small cube, respectively. N_v, N_h, N_d, N_s, and N are values derived from B_v, B_h, and B_d. A is the set of random numbers generated from a pseudo random number generator ($PRNG$) initialized with a seed δ. The starting point of the rotation (D, Y, X) is calculated from each number of A at t. The rotation direction Dir and rotation angle Ang are obtained from the random number at the time. % is the remainder operator; e.g., $num_1 \,\%\, num_2$ means num_1 modulo num_2. According to Dir, the cube is rotated ($\circlearrowleft$) depth-wise ($Dir = 0$), horizontally ($Dir = 1$), and vertically ($Dir = 2$).

The cube scrambling is illustrated in Fig. 4. It exchanges the positions of pairs of cubes corresponding to each random number. The operation is applied to all cubes. The details are described in Algorithm 2. The first step is initialization of $N_v, N_h, N_d, N_s, N, PRNG$, and A. The function derive($\cdot$)

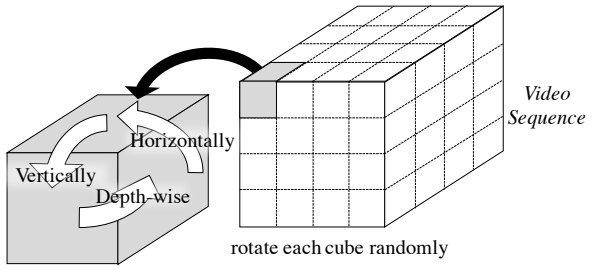

Fig. 3. Cube rotation.

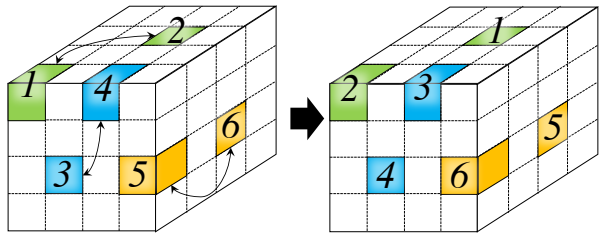

Fig. 4. Cube scrambling.

means that the variables are calculated by using Algorithm 1 and a random number $A(t)$ $(t = 0, 1, \cdots, N-1)$. The operator $\leftrightarrows$ means the exchange of two cubes.

The cube negative/positive transformation and cube color component shuffling are almost the same as conventional block-based case. To facilitate the discussion, we omit the explanation here and will not mention the two methods further.

B. Robustness against Illegal Decryption

The cube-based ETC system is more robust against illegal decryption than the block-based ETC system. In contrast with

Algorithm 1 Cube rotation

Input: *Video sequence* V_{in}
Output: V_{in} encrypted by rotating each small cube

$N_v \leftarrow$ height of each frame$/B_v$
$N_h \leftarrow$ width of each frame$/B_h$
$N_d \leftarrow$ number of frames$/B_d$
$N_s \leftarrow N_v \times N_h$
$N \leftarrow N_v \times N_h \times N_d$ //Number of cubes
$PRNG \leftarrow \delta$
$A \leftarrow$ arrange$\{a_0, \cdots, a_{N-1}; a_i \in [0, 1, \cdots, N-1]\}$
for $t = 0$ to $N-1$ **do**
  $X \leftarrow (A(t)\%N_s)\%N_h \times B_h$
  $Y \leftarrow (A(t)\%N_s)/N_h \times B_v$
  $D \leftarrow \lfloor A(t)/N_s \rfloor \times B_d$
  $Dir \leftarrow rand()\%3$ //horizontally, vertically, or depth-wise
  $Ang \leftarrow rand()\%4$ //$0°, 90°, 180°, 270°$
  if $Dir = 0$ **then**
    // Depth-wise rotate
    for $v = Y$ to $Y + B_v$ **do**
      $V_{in}(D : D + B_d, v, X : X + B_h, :) \circlearrowleft 90 \times Ang°$
    end for
  else if $Dir = 1$ **then**
    // Horizontally rotate
    for $d = D$ to $D + B_d$ **do**
      $V_{in}(d, Y : Y + B_v, X : X + B_h, :) \circlearrowleft 90 \times Ang°$
    end for
  else if $Dir = 2$ **then**
    // Vertically rotate
    for $h = X$ to $X + B_h$ **do**
      $V_{in}(D : D + B_d, Y : Y + B_v, h, :) \circlearrowleft 90 \times Ang°$
    end for
  end if
end for

Algorithm 2 Cube scrambling

Input: *Video sequence* V_{in}
Output: V_{in} encrypted by scrambling small cubes composing the video sequence

$\{N_v, N_h, N_d, N_s, N, PRNG, A\} \leftarrow$derive(**Algorithm 1**)
$t \leftarrow 0$
while $t \leq N-1$ **do**
  $(D_1, Y_1, X_1) \leftarrow$derive(**Algorithm 1**, $A(t)$)
  $(D_2, Y_2, X_2) \leftarrow$derive(**Algorithm 1**, $A(t+1)$)
  $V_{in}(D_1 : D_1 + B_d, Y_1 : Y_1 + B_v, X_1 : X_1 + B_h, :) \leftrightarrows V_{in}(D_2 : D_2 + B_d, Y_2 : Y_2 + B_v, X_2 : X_2 + B_h, :)$
  $t \leftarrow t + 2$
end while

the block-based case, the cube-based case, which utilizes multiple frames simultaneously, can exchange the blocks of each frame with the blocks of the other frames or spatiotemporal slices. This means that the illegal decryption from only each frame is clearly impossible.

However, new methods of decrypting encrypted frames illegally may be developed someday, e.g., a brute-force attack using each cube as a unit. Here, we can evaluate new methods by calculating the total number of brute-force attacks. The cube rotation is decrypted by rotating each cube in all directions and angles. Since there are three directions and four angles, each cube produces a total of twelve rotation variations. The maximum number of brute-force attacks against cube rotation for N cubes is

$$N_{cr} = 12^N.$$

The cube scrambling is decrypted by moving each cube back to its proper position. The probability that the third party can move one in N cubes back to its proper position is $\frac{1}{N}$ and the probability of it moving back one more cube properly is $\frac{1}{N(N-1)}$. Viewed in this way, the probability of it moving all cubes properly is

$$P_{cs} = \prod_{i=0}^{N-1} \frac{1}{N-i} = \frac{1}{N!}.$$

The maximum number of brute-force attacks against cube scrambling is

$$N_{cs} = N!.$$

Consequently, the total number of brute-force attacks against the two cube-based methods is

$$N_{ct} = N_{cr} \times N_{cs} = 12^N \times N!.$$

For example, when the cube rotation and cube scrambling are applied to a $288 \times 352 \times 256$ video sequence, N_{ct} is $12^{6336} \times 6336!$ in the case of $16 \times 16 \times 16$ cube. This number is much bigger than the 2^{256} of the brute-force attacks against the SHA-256 hash digest [11], and it is clear that attacks against these methods in real time are difficult.

IV. Experimental Results

We compared the compression quality of frames encrypted by the ETC systems with that of the unencrypted frames. Three raw video sequences *Akiyo*, *Bowing*, and *Coastguard* [12] were used. Each video sequence had a size of $288 \times 352 \times 256$ and 8-bit RGB full color space. The size of the divided small cubes was $16 \times 16 \times 16$ pixels in accordance with the conventional block-based case. We experimented as follows:

1) Encrypt the frames of a raw video sequence.
2) Compress the encrypted frames with *libjpeg* [13], whose compression qualities are $Q = \{10, 20, \cdots, 100\}$.
3) Calculate mean bitrates of the compressed frames.
4) Decompress the compressed frames.
5) Decrypt the decompressed frames.
6) Calculate mean PSNRs of the decrypted frames.

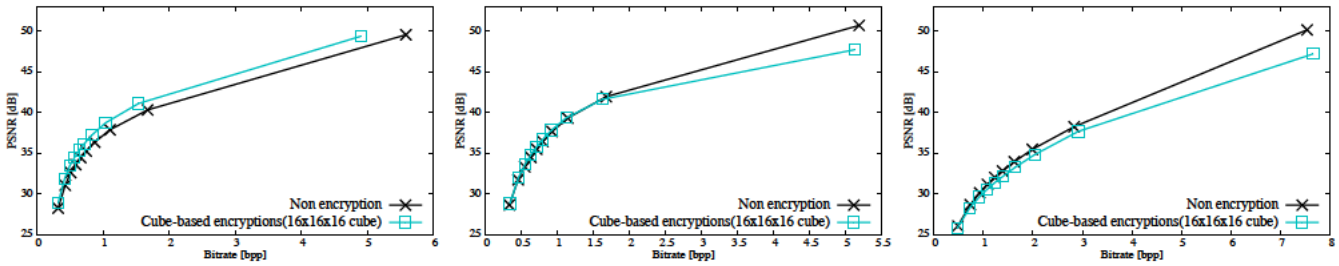

Fig. 5. Rate-distortion curve (mean of 256 frames): (a) *Akiyo*, (b) *Bowing*, (c) *Coastguard*.

Fig. 6. Visibility comparison of 140th frame of *Coastguard*: (left-to-right) original, only cube rotation, only cube scrambling, and cube rotation and scrambling.

The experimental results are shown in Fig. 5. The quality in encrypted *Akiyo* became better than the unencrypted one. The quality in encrypted *Bowing* was almost same as the unencrypted one except for $Q = 100$. The quality in encrypted *Coastguard* became worse than the unencrypted one. These mean that the proposed encryption influences Motion JPEG algorithm. We consider the reason is that the encrypted frame containing many almost invariant textures in different frames can be compressed efficiently by JPEG, whereas one containing other textures cannot be compressed so efficiently.

We also confirmed that the visibilities of the encrypted frames were reduced to enhance security. A visibility comparison of the 140th frame of *Coastguard* is shown in Fig. 6. In the case of only cube rotation, some of the blocks are not in the original frame. In the case of only cube scrambling and cube rotation and scrambling, the blocks are at different positions from those of the original frame. It is clear that we cannot recognize the original frames in the case of cube rotation and scrambling.

V. Conclusion

We proposed a cube-based ETC system for Motion JPEG. The cube-based encryption makes the illegal decryption from only each frame of a video sequence impossible and is robust against a brute-force attack. Experiments showed that the proposed encryption method affected the Motion JPEG in quality, whereas the level of encryption was high both objectively and subjectively.

Acknowledgment

This work was supported by a JSPS Grant-in-Aid for Young Scientists (B), Grant Number 16K18100.

References

[1] N. G. Bourbakis, "Image data compression-encryption using G-scan patterns," in *Proc. of ICSMC'97*, Orlando, FL, Oct. 1997, pp. 1117–1120.
[2] H. Cheng and X. Li, "Partial encryption of compressed images and videos," *IEEE Trans. Signal Process.*, vol. 48, no. 8, pp. 2439–2451, Aug. 2000.
[3] M. Ito, N. Ohnishi, A. Alfalou, and A. Mansour, "New image encryption and compression method based on independent component analysis," in *Proc. of ICTTA'08*, Damascus, Syria, Apr. 2008, pp. 1–6.
[4] D. Maheswari and V. Radha, "Secure layer based compound image compression using XML compression," in *Proc. of ICCIC'10*, Coimbatore, India, Dec. 2010, pp. 494–498.
[5] *Advanced Encryption Standard (AES)*, FIPS Pub. 197, Nov. 2001.
[6] A. Kingston, S. Colosimo, P. Campisi, and F. Autrusseau, "Lossless image compression and selective encryption using a discrete Radon transform," in *Proc. of ICIP'07*, San Antonio, TX, Sep. 2007, pp. 465–468.
[7] F. Ahmed, M. Y. Siyal, and V. U. Abbas, "A perceptually scalable and JPEG compression tolerant image encryption scheme," in *Proc. of PRIVT'10*, Singapore, Nov. 2010, pp. 232–238.
[8] O. Watanabe, T. Fukuharai, and H. Kiya, "A perceptual encryption scheme for Motion JPEG 2000 standard," in *Proc. of ISCIT'15*, Nara, Japan, Oct. 2015, pp. 125–128.
[9] K. Kurihara, M. Kikuchi, S. Imaizumi, S. Shiota, and H. Kiya, "An encryption-then-compression system for JPEG/Motion JPEG standard," *IEICE Trans. Fundamentals*, vol. E98-A, no. 11, pp. 2238–2245, Nov. 2015.
[10] T. Chuman, K. Kurihara, and H. Kiya, "On the security of block-based ETC systems against jigsaw puzzle solver," in *Proc. of ICASSP'17*, New Orleans, LA, Mar. 2017, pp. 2157–2161.
[11] *Secure Hash Standard (SHS)*, FIPS Pub. 180-4, Mar. 2012.
[12] "Xiph.org video test media [derf's collection]," Available: https://media.xiph.org/video/derf/.
[13] "JPEG software," Available: https://jpeg.org/jpeg/software.html.