On the Security of Bitstream-level JPEG Encryption with Restart Markers

Mare Hirose*, Shoko Imaizumi* and Hitoshi Kiya†
* Chiba University, Chiba, Japan
E-mail: marehirose@chiba-u.jp, imaizumi@chiba-u.jp
† Tokyo Metropolitan University, Tokyo, Japan
E-mail: kiya@tmu.ac.jp

***Abstract*—This paper aims to evaluate the safety of a bitstream-level JPEG encryption method with restart (RST) markers, in which encrypted data can preserve the JPEG file format with the same size as that without encryption. Data encrypted with the method can be decoded without any modification of header information by using a standard JPEG decoder. In addition, the use of RST markers allows us to define extended blocks separated by the RST markers, so spatially partial encryption and block-permutation-based encryption can be carried out. However, the security of the method was only evaluated in terms of the key space analysis for brute-force attacks and other limited attacks. Accordingly, in this paper, we perform the security evaluation of the method in terms of robustness against ciphertext-only attacks including state-of-the-art attacks. In experiments, the method is compared with conventional encryption methods, and it is verified to be robust against ciphertext-only attacks if parameters used for image encryption are carefully chosen.**

I. Introduction

The use of JPEG images has increased due to the growth of social networking services. These services do not allow the use of arbitrary file formats in general. In many cases, images have to have compliance with the JPEG standard. In addition, these services are not reliable due to incidents such as information leakage. Accordingly, many encryption methods that preserve the JPEG format have been proposed so far. Recently, a bitstream-level encryption method for JPEG compression was proposed, and it was verified to outperform conventional encryption methods in some important respects [1]. However, it is not verified enough yet in terms of the robustness of the encryption against various attacks. Accordingly, we aim to evaluate the security of the method against ciphertext-only attacks (COAs) in this paper.

Bitstream-level encryption is one of types for encryption combined with image compression. Compared with other types, this type of encryption has some advantages. For example, the type allows us not only to maintain the JPEG file format but also to use standard JPEG encoders. In addition, some methods of this type ensures that the file size remains the same before and after encryption, so encryption can be carried out by hooking images within a transmission channel. A state-of-the-art method [1] can also generate partially encrypted images, but all other bitstream-level methods cannot do that.

In [1], the restart (RST) marker, inserted at regular intervals between minimum coded units (MCUs), is used for encryption. This marker contributes to the mixing of encrypted and unencrypted regions. Furthermore, MCUs separated by RST markers can be defined as extended blocks, and the visibility of the encrypted image can be reduced by permuting the position of these blocks. This block permutation is also expected to enhance security strength.

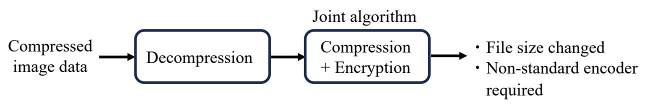
Accordingly, we focus on the state-of-the-art method to evaluate the attack resistance of the method against ciphertext-only attacks (COAs) in this paper. Key space analysis, key sensitivity analysis, the non-zero counting attack (NZCA) and histogram analysis are performed, and the relationship between the use of RST markers and the attack resistance is discussed. In experiments, it is demonstrated that the RST marker that the method used in image encryption for the first time can enhance the robustness against various attacks in general, and the attack resistance of encrypted images depends on the selection of restart interval values.

II. Related Work

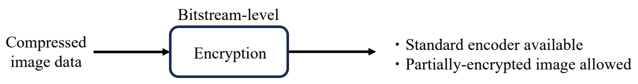
The encryption method discussed in this paper is a method combined with image compression that can generate JPEG files with hidden visual information [1]. Accordingly, previous encryption methods combined with image compression are summarized here. In addition, the structure of JPEG bitstreams is briefly explained.

A. Combined Use of Encryption and Image Compression

Encryption methods combined with image compression are classified into the four types as below. Type 1 is compression-then-encryption with standard cryptography, in which images are compressed and then encrypted with a standard cryptography such as AES (advanced encryption standard). Type 2 is encryption-then-compression, in which the visual information of images is protected by using a perceptual encryption method, called a compressible encryption method, and then the encrypted images are compressed [2], [3]. Type 3 is combining encryption and compression. For type 3, encryption and compression are simultaneously carried out (see Fig. 1(a)) [4]–[7]. Thus, given JPEG images prior to encryption, the JPEG images must be decompressed before carrying out encryption. In addition, standard encoders cannot be used in this framework. Type 4 is compression-then-encryption with a bitstream-level encryption method (see Fig. 1(b)), in which



(a) Type 3


Bitstream-level
Compressed image data
Encryption
・Standard encoder available
・Partially-encrypted image allowed

(b) Type 4

Fig. 1. Types of encryption.

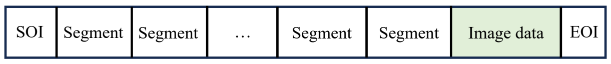


Fig. 2. Structure of JPEG bitstream.

images are compressed and then encrypted with a bitstream-level encryption method [1], [8]–[11]. Type 4 encryption allows us not only to maintain the JPEG format but to also use standard JPEG decoders.

We focus on considering a state-of-the-art method [1], which is a type 4 encryption method for JPEG images. The method enables us to generate encrypted images with various visibility, including partially encrypted images.

B. JPEG bitstream and marker codes

Here, we describe the structure of JPEG bitstream. Fig. 2 shows an example of the structure of JPEG bitstreams [13]. The codes called SOI and EOI, which signify the beginning and end of a bitstream respectively, are allocated. These are two-byte codes called marker codes. Segments contain information necessary for decoding, such as Huffman tables and quantization tables. Marker codes are also allocated at the beginning of each segment. The first byte of the marker code is fixed at $FF_{(16)}$, and the second byte indicates the type of each marker. The second byte of marker code distinguishes each segment. Note that $FF00_{(16)}$ is not defined as a marker code.

Fig. 3 shows the structure of image data of a JPEG bitstream. This shows the case of 4:2:0 color subsampling. Image data is divided and processed in units called minimum coded units (MCUs). Each MCU consists of four luminance components (Y) and two chroma components (Cb, Cr). Each component contains a DC coefficient and AC coefficients, and the DC coefficient is recorded as the difference value from the previous DC coefficient. Finally, these bit sequences are divided and stored byte by byte. Due to this process, $FF_{(16)}$ may occur in image data. Therefore, the JPEG encoder inserts $00_{(16)}$ immediately after $FF_{(16)}$ occurs in image data to distinguish it from the first byte of a marker code. Additionally, when the JPEG decoder detects $FF00_{(16)}$, it reads only $FF_{(16)}$ and skips $00_{(16)}$. This process is called "byte stuffing".

One of the JPEG marker codes is called the RST marker, which can be inserted at regular intervals among MCUs. The restart interval (RI), which is an interval at which the RST marker is inserted, is set during JPEG encoding. When a RST marker is not used, an error that occurs in the DC coefficient in a bitstream propagates to the end of a bitstream. On the other hand, the value of a DC coefficient immediately after a RST marker is stored as its original value. Therefore, using RST markers prevents errors from propagating among DC coefficients in bitstreams.

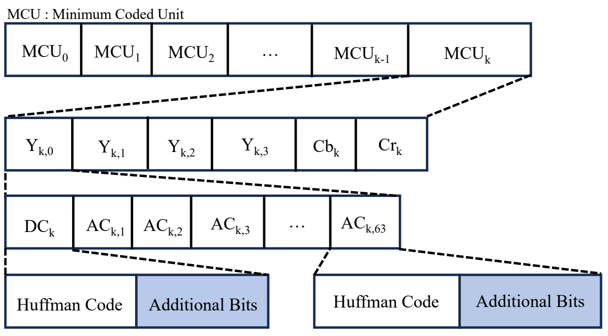


Fig. 3. Structure of image data.

III. Security Evaluation

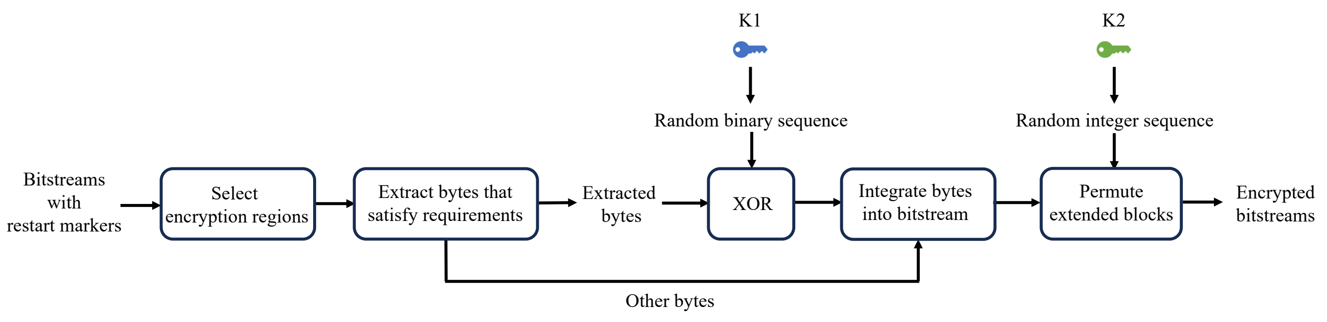
A. Bitstream-level JPEG Encryption with RST Markers

The bitstream-level JPEG encryption method with RST markers [1] is summarized here. Fig. 4 shows the encryption process of the method [1]. The method utilizes RST markers for encryption, where a series of MCUs separated by RST markers is referred to as an extended block (see Fig. 5). The encryption procedure for this method is shown below.

Step 1: Select an RI and prepare a bitstream including RST markers to be encrypted.
Step 2: Select encryption regions.
Step 3: Extract bytes that satisfy encryption conditions from the extended blocks to maintain the same file size as that before encryption.
Step 4: Generate a pseudorandom (PRN) binary sequence by using secret key K1. Carry out an exclusive-or (XOR) operation between the PRN sequence and additional bits of the extracted bytes.
Step 5: Replace the additional bits with the XOR results.
Step 6: Randomly permute the positions of the extended blocks defined in Step 1 using secret key K2.

The encryption conditions in Step 3 were discussed in [11], [12]. In [11], [12], the bytes of a bitstream are categorized into five cases, as shown in Fig. 6, where the white, blue, and red areas represent Huffman codes, additional bits, and byte placed immediately after $FF_{(16)}$, respectively. Each case is defined below.

Case 1: It consists of only Huffman codes.
Case 2: It consists of only additional bits.
Case 3: It consists of Huffman codes and additional bits, and every bit in the Huffman code is 1.
Case 4: It consists of Huffman codes and additional bits, and the Huffman code includes 0.



K1
K2
Random binary sequence
Random integer sequence
Bitstreams with restart markers
Select encryption regions
Extract bytes that satisfy requirements
Extracted bytes
XOR
Integrate bytes into bitstream
Permute extended blocks
Encrypted bitstreams
Other bytes

Fig. 4. Block diagram of encryption process.

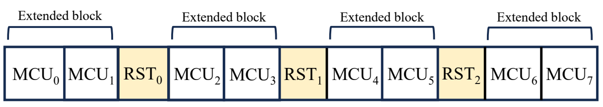


Fig. 5. Bitstreams after insertion of RST markers (RI = 2).

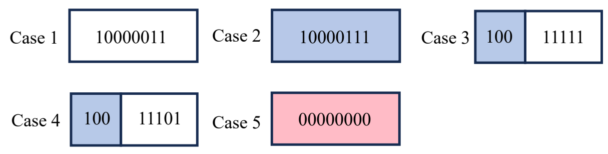


Fig. 6. Five types of bytes.

Case 5: It consists of 0 only, and the byte is placed immediately after FF$_{(16)}$.

Among these cases, only additional bits in Case 4 can be encrypted.

In [1], each extended block is encrypted separately. Thus, it is possible to generate a partially encrypted image that contains both encrypted and unencrypted regions. Fig. 7 shows an example of images encrypted with this method.

B. Threat Model

A threat model includes a set of assumptions, such as an adversary's goals, knowledge, and capabilities. The objective of an attacker is to restore visual information from encrypted images. We assume that the attacker has access to encrypted images and the encryption algorithm but does not possess the secret key. Accordingly, the attacker can only carry out ciphertext-only attacks (COAs) using encrypted images.

C. Security Analysis

Several COAs have been proposed to restore visual information from encrypted images [4], [9], [10], [12]. In addition, security analysis methods are used to evaluate the robustness of encryption methods against attacks in general. In this paper, we use key space analysis, key sensitivity analysis, the non-zero counting attack (NZCA), and histogram analysis. The relationship between the use of RST markers and the attack resistance is discussed.

1) Key space analysis: Key space is the total number of patterns that can be generated by a given encryption algorithm. Here, we analyze the encryption method in the case where all DCT coefficients in the entire image are encrypted. The algorithm encrypts additional bits within bytes that satisfy specific conditions and permutes the positions of the extended blocks, as shown in III-A. We suppose that JPEG encoding is applied to an $M \times N$-pixel image with $RI =$ r and 4:2:0 color subsampling. First, considering that the number of bytes to be encrypted is T, the minimum and maximum sizes of the key space derived from the encryption of the additional bits S_{enc_min} and S_{enc_max} are expressed as follows:

$$S_{enc\_min} = 2^T, \tag{1}$$
$$S_{enc\_max} = 2^{7T}. \tag{2}$$

Note that the minimum number of bits to be encrypted is one, and the maximum is seven within a byte that satisfies the encryption conditions. The key space obtained from the permutation of the extended blocks, S_{bp}, is expressed as

$$S_{bp} = \left\lfloor \left(\left\lceil \frac{M}{16} \right\rceil \times \left\lceil \frac{N}{16} \right\rceil \times \frac{1}{r} \right) \right\rfloor !. \tag{3}$$

Accordingly, the overall minimum key space S_{min} and maximum key space S_{max} are given by

$$\begin{aligned} S_{min} &= S_{enc\_min} \times S_{bp} \\ &= 2^T \times \left\lfloor \left(\left\lceil \frac{M}{16} \right\rceil \times \left\lceil \frac{N}{16} \right\rceil \times \frac{1}{r} \right) \right\rfloor !, \end{aligned} \tag{4}$$

$$\begin{aligned} S_{max} &= S_{enc\_max} \times S_{bp} \\ &= 2^{7T} \times \left\lfloor \left(\left\lceil \frac{M}{16} \right\rceil \times \left\lceil \frac{N}{16} \right\rceil \times \frac{1}{r} \right) \right\rfloor !. \end{aligned} \tag{5}$$

Equation (3) shows that the number of extended blocks increases as RI is shorter. Therefore, the encrypted image with a shorter RI has a larger key space. The algorithm with RST markers that can use a shorter RI can provide a larger key space. For example, if the image in Fig 7(a) is encoded with r = 4 as an RI, T = 37,031 is given. Thus, the minimum key space S_{min} can be expressed by

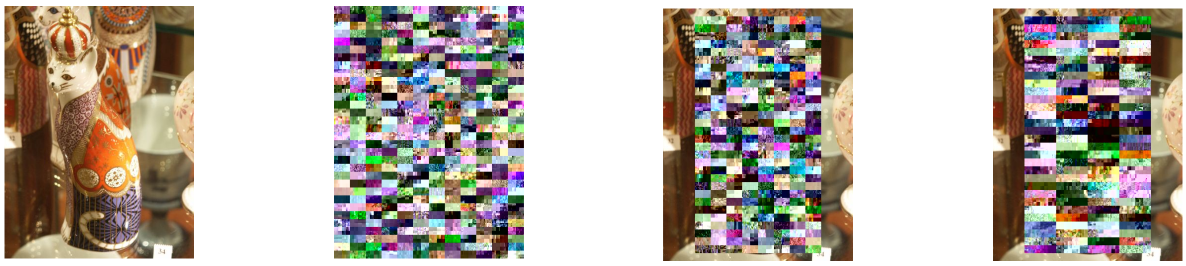

(a) Original image (b) Encrypted image (RI =2) (c) Partially-encrypted image (RI =2) (d) Partially-encrypted image (RI =4)

Fig. 7. Examples of encrypted images (ucid00459).

$$S_{min} = 2^{37,031} \times \left\lfloor \left(\left\lceil \frac{384}{16} \right\rceil \times \left\lceil \frac{512}{16} \right\rceil \times \frac{1}{4} \right) \right\rfloor ! > 2^{256}. \quad (6)$$

It is enough larger than 2^{256}, which is the space of a key with 256 bits. If we use $r = 2$, S_{min} is larger than that of $r = 4$. Accordingly, the bitstream-level encryption method with RST markers has enough key space.

2) Key sensitivity analysis: Attack-resistant encryption methods should be sensitive even to a slight change in the key. We analyze key sensitivity by considering the following two cases:

Case 1: Encryption using an encryption key that differs from the original key by only one bit.

Case 2: Decryption using an incorrect key that differs from the correct key by only one bit.

In Case 1, the encrypted image must be completely different from the image encrypted with the original key. In Case 2, the image decrypted with an incorrect key must be completely different from the original image.

3) NZCA: The sketch attack tries to obtain contour information of the original image from the encrypted image. We use the non-zero-counting attack (NZCA), which is a sketch attack for encrypted JPEG images [4]. This attack attempts to restore the original image's contours based on the number of non-zero coefficients in each MCU of the encrypted image. The use of RST markers will be evaluated to generate robust encrypted images against NZCA.

4) Histogram analysis: The histogram is very useful for an adversary to analyze the data distribution graphically in the encrypted domain. An adversary tries to analyze the distribution of pixel frequencies in encrypted images to apply attacks. Accordingly, attack-resistant encryption methods should provide encrypted images with histograms similar to those of other encrypted images.

IV. Experimental Results

A. Setup

In experiments, we used 1,338 test images with 384 × 512 pixels or 512 × 384 pixels from Uncompressed Colour Image Database [14]. JPEG compression was performed using libjpeg [15], with the color subsampling set to 4:2:0 and the quality factor Q set to 80. Pseudo-random number (PRN) binary sequences were generated using HMAC_DRBG [16] with a 384-bit key. For all images, we encrypted the entire image and permuted the positions of all extended blocks.

B. Protection of visual information

The encryption method with RST markers can randomly permute the positions of blocks, thereby strongly protect the visual information of images, compared with previous methods without block permutation [5], [11], [12]. To objectively evaluate the protection strength, peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) values of encrypted images were shown as boxplots in Fig.8. From the figure, it is confirmed that the method generates encrypted images without identifiable visual information.

C. Key sensitivity

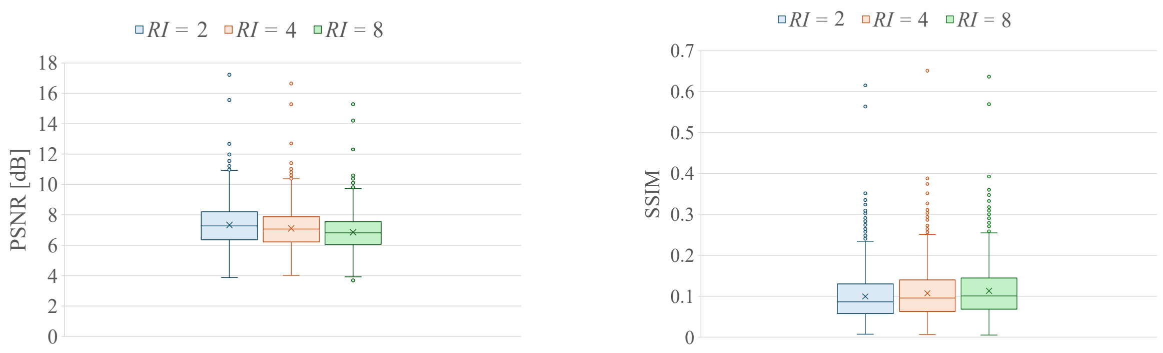
Fig. 9 illustrates the results of key sensitivity analysis where SSIM values indicates the difference between two images. In Fig. 9(a), two images were encrypted with keys that differed by a single bit. In contrast, in Fig. 9(b), the two images were decoded ones from an encrypted one by using the correct key and an incorrect key that differed from the correct key by a single bit. From the figure, the method was verified to have high key sensitivity enough.

D. NZCA

NZCA was performed on the luminance component of the encrypted image. As shown in Fig. 10, in the previous method without RST markers [11], [12], the original image was revealed in its outline. In comparison, the method with RST markers concealed the original image. This was achieved by replacing extended blocks. However, when a large RI was adopted, the outline of the original image was revealed in some areas. Accordingly, the use of RST markers is important for enhancing robustness against attacks, but the value of RI should be carefully selected.

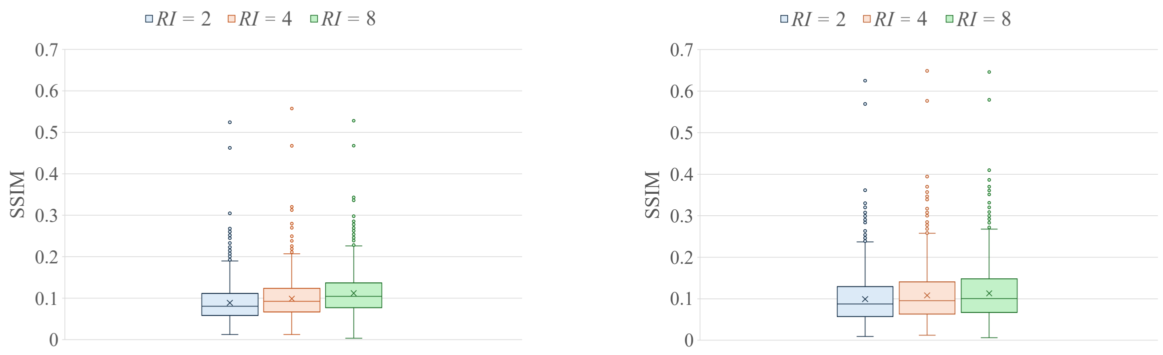
E. Histogram analysis

Fig. 11 illustrates the R, G, and B histograms of the original ucid00459 image, which is shown in Fig. 7(a), and its encrypted images. In the figure, there exist the histograms for four encrypted images. Three of them were derived by



(a) PSNR (b) SSIM

Fig. 8. Evaluation of visual protection. The top of the box, the bottom of the box, and the line in the middle represent the 75th percentile, the 25th percentile, and the 50th percentile, respectively. The whiskers denote the highest and lowest values that are not outliers. Circles are outliers, and crosses are the means.



(a) Encryption with different keys (Case 1) (b) Decryption with incorrect keys (Case 2)

Fig. 9. Key sensitivity analysis results. The top of the box, the bottom of the box, and the line in the middle represent the 75th percentile, the 25th percentile, and the 50th percentile, respectively. The whiskers denote the highest and lowest values that are not outliers. Circles are outliers, and crosses are the means.

using our method with RST markers and the other was derived by using the previous method. In our method with RST markers, RI was defined as 2, 4, and 8. The images encrypted using the method with RST markers were compared with the original image and those encrypted with the previous method. Attack-resistant encryption methods should provide an encrypted image with a totally different histogram from that of the original image and analogous histograms among three color channels. From the figure, we can see that R, G, and B channels in the encrypted image have analogous histograms. In addition, the use of a small RI can strongly reduce the characteristics of the image. Other images were verified to have similar trends.

V. Conclusion

In this paper, we conducted security analyses on state-of-the art encryption method with RST markers against ciphertext-only attacks. It was confirmed that RST markers used in the method contribute to enhance security in terms of key space, key sensitivity, NZCA, and histogram analysis. As a result, the method was demonstrated not only to outperform conventional JPEG encryption methods in some important respects but to also maintain a high security level. Furthermore, it was shown that attack resistance varies with the length of the restart interval.

References

[1] M. Hirose, S. Imaizumi, and H. Kiya, “Encryption Method for JPEG Bitstreams for Partially Disclosing Visual Information,” *Electronics,* vol. 13, no. 11, 2024.

[2] W. Sirichotedumrong, T. Chuman, S. Imaizumi and H. Kiya, “Grayscale-Based Block Scrambling Image Encryption for Social Networking Services,” in *Proc. IEEE Int. Conf. Multimed. Expo,* pp. 1–6, 2018.

[3] H. Kiya, A. P. M. Maung, Y. Kinoshita, S. Imaizumi and S. Shiota, “An overview of compressible and learnable image transformation with secret key and its applications,” *APSIPA Trans. Signal Inf. Process.,* vol. 11, no. 1, 2022.

[4] Y. Peng, C. Fu, G. Cao, W. Song, J. Chen and C. W. Sham, “JPEG-compatible Joint Image Compression and Encryption Algorithm with File Size Preservation,” *ACM Trans. Multimed. Comput. Commun. Appl.,* vol. 20, no. 4, pp. 1–20, 2024.

[5] K. Shimizu and T. Suzuki, “Finely Tunable Bitcuboid-based Encryption with Exception-free Signed Binarization for Jpeg Standard,” *IEEE Trans. Inf. Forensics Secur.,* vol. 16, pp. 4895–4908, 2021.

[6] Cao. W, Leng. X, Yu. T, Gu. X and Liu. Q, “A Joint Encryption and Compression Algorithm for Multiband Remote Sensing Image Transmission,” *Sensors,* vol. 23, no. 17, 2023.

[7] O. Watanabe, A. Nakazaki and H. Kiya, “A fast image-scramble method using public-key encryption allowing backward compatibility with JPEG2000” in *Proc. Int. Conf. Image Process.,* pp. 3435–3438, 2004.

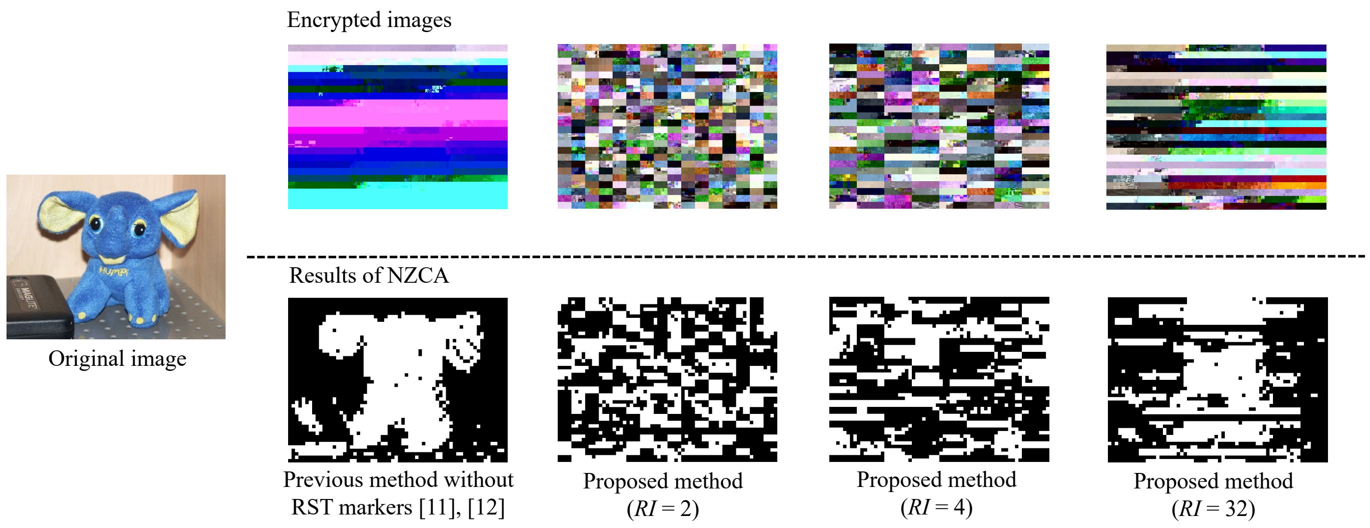

Encrypted images
Original image
Results of NZCA
Previous method without RST markers [11], [12]
Proposed method ($RI = 2$)
Proposed method ($RI = 4$)
Proposed method ($RI = 32$)

Fig. 10. Results of non-zero-counting attack (ucid00294).

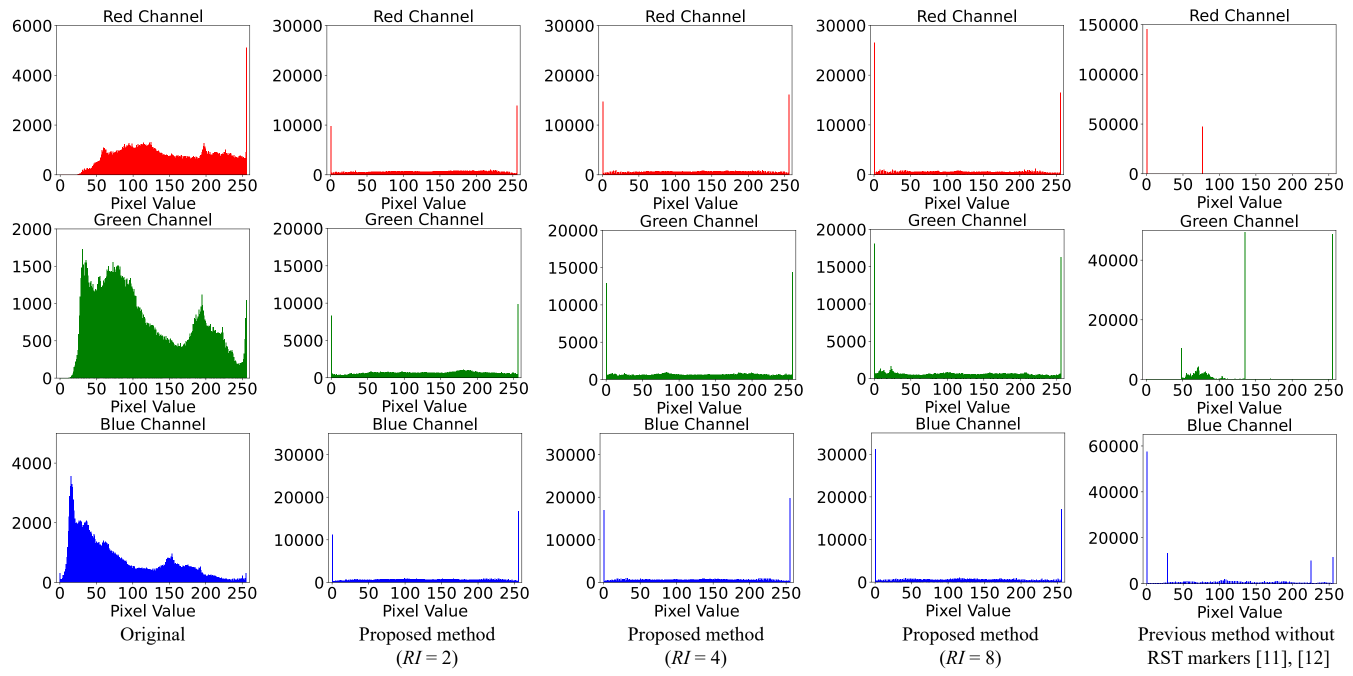


Fig. 11. Histogram Analysis (ucid00459).

[8] Y. Yuan, H. He, Y. Yang, N. Mao, F. Chen and M. Ali, "JPEG Image Encryption with Grouping Coefficients Based on Entropy Coding," *J. Vis. Commun. Image Represent.,* vol. 97, 2023.

[9] J. He, S. Huang, S. Tang and J. Huang, "JPEG Image Encryption With Improved Format Compatibility and File Size Preservation," *IEEE Trans. Multimed.,* vol. 20, no. 10, pp. 2645–2658, 2018.

[10] C. Qin, J. Hu, F. Li, Z. Qian and X. Zhang, "JPEG Image Encryption with Adaptive DC Coefficient Prediction and RS Pair Permutation," *IEEE Trans. Multimed.,* vol. 25, pp. 2528–2542, 2022.

[11] H. Kobayashi and H. Kiya, "Bitstream-based Jpeg Image encryption with File-size Preserving," in *Proc. IEEE Glob. Conf. Consum. Electron.,* pp. 384–387, 2018.

[12] H. Kobayashi and H. Kiya, "File-Size Preserving Encryption of JPEG Images in the Bitstream Domain," *IEICE Trans. Inf.& Syst. Jpn. Ed.,* vol. J102-D, no. 12, pp. 787–795, 2019.

[13] *Information technology—Digital Compression and Coding of Continuous-Tone still Images:Requirements and Guidelines.* International Organization for Standardization ISO/IEC IS-10918-1, 1994.

[14] G. Schaefer and M. Stich, "UCID: An Uncompressed Color Image Database," in *Proc. SPIE,* vol. 5307, pp. 472–480, 2003.

[15] [Online] Available:URL:https://www.ijg.org

[16] Barker, E. B. and Kelsey, J. M, *SP 800-90A. Recommendation for Random Number Generation Using Deterministic Random Bit Generators.* NIST: Gaithersburg, MD, USA, 2012.