ENCRYPTION AND DATA INSERTION TECHNIQUE USING REGION DIVISION AND HISTOGRAM MANIPULATION

Ryoma Ito[1], KokSheik Wong[2*], Simying Ong[3], and Kiyoshi Tanaka[4]

[1]Faculty of Engineering, Shinshu University, Japan.

[2]School of Information Technology, Monash University Malaysia, Malaysia.

[3]Faculty of Computer Science & Information Technology, University of Malaya, Malaysia.

[4]Academic Assembly (Institute of Engineering), Shinshu University, Japan.

Email: ryoma.ito.316@gmail.com[1], wong.koksheik@monash.edu[2*] (corresponding author), simying.ong@um.edu.my[3], ktanaka@shinshu-u.ac.jp[4]

DOI: https://doi.org/10.22452/mjcs.vol34no4.2

ABSTRACT

Traditionally, encryption is applied to mask the semantic of a content, while data insertion adds data to a content for management purpose. In this work, a joint encryption and data insertion method is proposed. The input image is divided into 2 parts, where the first part is manipulated to mask the perceptual semantics, while the second part is processed to hide data. The binary image, which is the data to be inserted, further divides the second part of the input image into 2 regions, one being the 'zero' region and the other being the 'one' region. Pixels of the original image at position coinciding with the 'zero' region are darken, while those coinciding with the 'one' region are brightened. The darkening and brightening processes are performed by using histogram matching technique. Furthermore, two techniques are introduced to improve robustness of the inserted binary image against basic image processing operations. The proposed joint method allows the inserted binary image to be extracted directly from the masked image or from the reconstructed image. The proposed method is also commutative because the same result is achieved irregardless of the order of processing in encrypting and inserting data.

Keywords: Region division, safety zone, JEDI

1.0 INTRODUCTION

Joint encryption and data insertion (JEDI) has received much attention in recent years thanks to the features it offers in addressing today's applications. For example, image is encrypted before transmission or uploaded to online storage to avoid unauthorized viewing, while data is inserted to facilitate the claim of ownership, inscribing fingerprint, authentication, management of content, hyperlinking, to name a few [1, 2, 3].

Over the years, there are many innovations in achieving JEDI. In addition to the sequential approach of encryption-then-insertion and insertion-then-encryption, one of the most common ways to achieve JEDI is to split the content into two non-overlapping parts, where each part is manipulated to achieve different objectives. For example in Zhang's method [4], five most significant bit (MSB) planes of an image is manipulated to encrypt the image, while the remaining 3 bit planes are divided into 2 groups, where one of the groups is flipped[1] to encode '1', or left as it is to encode '0'. An approximation of the original image can be obtained by considering the correlation among the least significant bit (LSB) planes. Another class of approaches is to insert data to purposely mask the image. One such approach is proposed by Ong et al. [1], where each pixel in some selected range is associated to another pixel value outside of the range. The ranges are defined in a way so that the difference between any pair of associated pixel values is large. The original value is assumed when '0' is to be inserted, while the associated value is output when '1' is to be inserted. *([1]Performing the operation $2^k - 1 - x$ for each $k-$bit pixel of value x in the selected group).*

Most related works aim to improve the embedding capacity by determining optimum location for data embedding [18, 19, 20]. Wu et al. [18] utilizes Median Edge Detector (MED) to first predict image pixels using their respective neighboring pixel values, then calculate the prediction errors by subtracting it with the original value. These prediction errors are used to determine the location of embeddable pixels and will be marked as extra information using a compression technique called parametric binary tree labelling to ensure high capacity and reversibility during the decoding process. Recently, Wang et al. [19] proposed to first encrypt the image within the separated non-overlapping

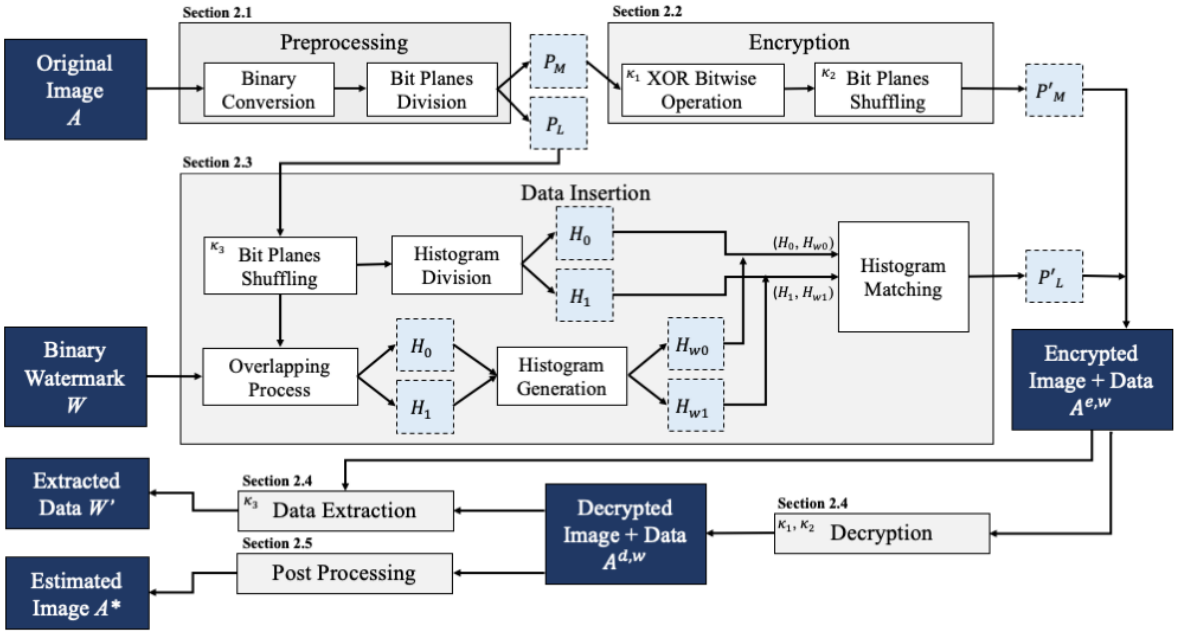


Fig. 1: The architectural framework of the proposed method illustrating the encoding and decoding processes.

blocks, then adaptively compress the pixels in the block by using the correlations of the image pixels to vacate spaces for secret bit insertion. Similarly, Yu et al. [20] proposed to use MED to identify the prediction error of the image bit-planes in a hierarchical manner and to determine the reversible (viz., predictable) bit-plane location in every image pixel for data hiding using the bit-plane replacement technique.

Departing from the spatial domain, researchers also transformed the input image into another domain. For example, Cancellaro et al. [5] transform the input image using Tree-Structure Haar transform, where the coefficients are decomposed into bit planes for further processing. Specifically, the MSB bit planes of the coefficients are encrypted by using advanced encryption system (AES), while the LSB bit planes are manipulated to encode data. JEDI is also realized in other domains such compressed image and video [6, 7, 8], where syntax elements of the compression standards are judiciously manipulated for format compliance.

While the conventional JEDI methods are able to achieve its objectives, the inserted data is usually fragile. For example, the inserted data is usually destroyed, even for commonly performed operations such as JPEG compression. Therefore, in this work, a JEDI method based on region division and histogram matching is proposed. Specifically, a few MSB bit planes are manipulated to mask the image, while the remaining LSB bit planes are modified to insert data. Notably, based on the binary image to be inserted, the pixels in the host image are darken when they coincide with the 'zeros' in the image, or brighten otherwise. Furthermore, two techniques are put forward to improve robustness against commonly performed image processing operations, which include lossy compression and low-pass filtering. The inserted data can be extracted from both the encrypted and decrypted image. Unless specified otherwise, we utilize the term encryption to refer to perceptual masking.

The rest of this paper is structured as follows: Section 2 puts forward the proposed JEDI method, and two strategies are presented in Section 3 to improve robustness against JPEG compression and low-pass-filtering. Experiment results are discussed in Section 4, and functional comparison between the proposed and conventional JEDI techniques are presented in Section 5. Finally, Section 6 concludes this paper.

2.0 PROPOSED JEDI METHOD

Figure 1 illustrates the architectural overview of the proposed JEDI method. The detailed information about each process is described in the following subsections. Without loss of generality, assume that the image A of dimension $M \times N$ has a bit-depth of 8. Let $A(x, y)$ refer to the pixel value at position (x, y) for $x \in [1, M]$ and $y \in [1, N]$.

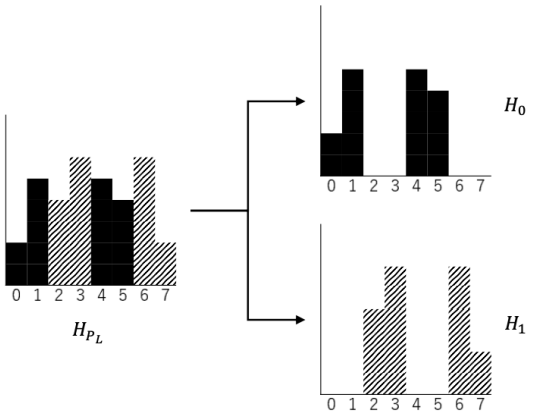


Fig. 2: Construction of H_0 and H_1 from H_{P_L} when $\tau = 3$ and $\varphi = 4$. Here, there are 4 sub-histogram clusters, and each cluster consists of 2 bins.

2.1 Pre-processing

Each pixel $A(x, y)$ in the image is converted into its respective binary values so that

$$A(x,y) = \underbrace{a_7 \parallel a_6 \parallel a_5 \parallel a_4}_{P_M} \underbrace{\parallel a_3 \parallel a_2 \parallel a_1 \parallel a_0}_{P_L} \quad (1)$$

where '‖' denotes to the concatenation operation. Here a_7 and a_0 are the most and least significant bits, respectively. The bit planes in each pixel are split into non-overlapping parts, namely, P_M for encryption purpose, and P_L for the data insertion purpose. The number of bit planes in P_L is controlled by the parameter τ, where $1 \leq \tau \leq 7$ and it is first defined. Subsequently, P_M has $(8 - \tau)$ bit planes.

2.2 Encryption

Two techniques are introduced to achieve encryption. The first technique uses the key κ_1 to generate a stream of randomized 0's and 1's, called keystream. This keystream is then utilized to perform the XOR operations on the bit planes in $\{P_M\}$ to generate $\{\hat{P}_M(x, y)\}$, which visually masks the image. Next, that resulting values of $\{\hat{P}_M(x, y)\}$ are shuffled (i.e., the locations are changed) by using another key κ_2 to obtain $\{P'_M(x, y)\}$ which further intensifies the masking effect. These 2 techniques lead to significant changes to the pixel value, and they can be repeated multiple rounds if required.

2.3 Data Insertion

First, the sub-image $\{P_L(x, y)\}$ is shuffled to obtain $P_L^*(x, y) = P_L(x', y')$. The position (x', y') are determined by using the key κ_3. The histogram $H_{P_L} = \{h[i]\}$ of P_L^* is the constructed, where $h[i]$ denotes the number of occurrences for pixel value i. Noted that the range of values for P_L is $[0, 2^\tau - 1]$. The resulting histogram is further split into two histograms, namely H_0 and H_1. In particular, the non-overlapping bins in H_0 and H_1 are gathered by using parameter ϕ, where $\phi = 2^q$, and the chosen parameter q value must be an integer between 1 and τ. Specifically, the histogram bins in H_{P_L} (viz., the total of $2^\tau - 1$ bins) are evenly divided into ϕ fragments. The odd fragments are collected to form H_0 while the remaining fragments (i.e., the even fragments) form H_1. For instance, when $\tau = 3$, the histogram bins are limited by the range of values in the sub-images P_L to $[0, 2^3 - 1]$. In Fig. 2, assuming that $q = 2$, then $\phi = 2^2 = 4$. Hence the histogram bins are divided into four non-overlapping fragments (viz., two bins per fragment in this example), which are later assigned to H_0 and H_1, alternatively.

Without loss of generality, suppose that the data to be inserted W is a binary image of $M \times N$ pixels, such that the value at position (x, y), denoted by $W(x, y)$, is either '0' or '1'. The shuffled set $P_L^*(x, y)$ is first overlaid with W and divided into two parts based on the values of W as follows:

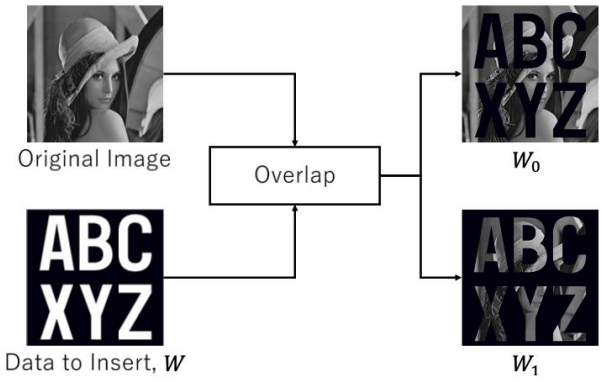


Fig. 3: Overlapping the watermark image with P_L. Here, the Lenna image is shown for illustration purposes.

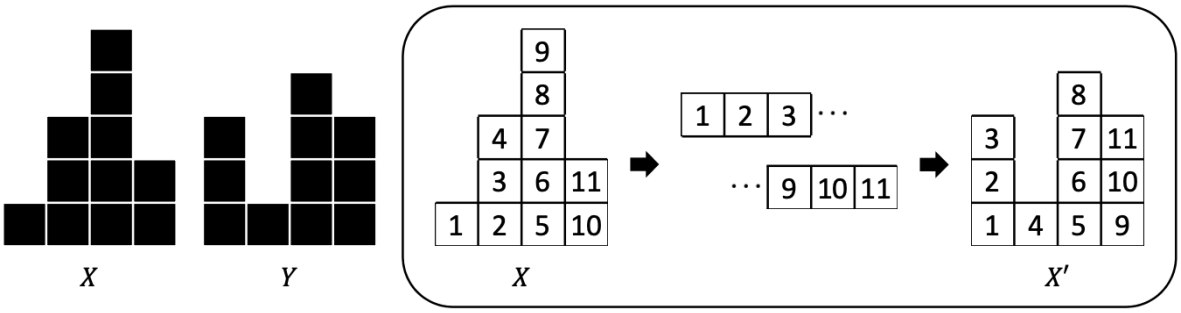


Fig. 4: Example of histogram shape conversion where histogram *X* is processed to assume the shape of histogram *Y*.

$$W_0(x,y) \leftarrow \big(1 - W(x,y)\big) \times P_L^*(x,y),$$
$$W_1(x,y) \leftarrow W(x,y) \times P_L^*(x,y). \tag{2}$$

An example is shown in Fig. 3. In W_0, if $W(x,y) = 1$, it will be replaced by the value $P_L^*(x,y)$, otherwise if $W(x,y) = 0$, it will be updated to value 1. Likewise, the opposite operation is performed to generate W_1. The histogram of W_0 and W_1 are constructed and denoted by H_{w0} and H_{w1}, respectively.

To encode data, exact histogram specification technique proposed by Coltuc et al. [9] is deployed for changing the shape of the histogram H_{w0} into the shape of the histogram H_0. Similarly, the shape of the histogram H_{w1} is transformed to that of H_1. With this approach, pixels in W_0 will be transformed into a relatively smaller value, shifting to the odd fragments location in H_0 (i.e., darken). Likewise, pixels in W_1 will be brighten slightly to match with histogram of the even fragments in H_1. Figure 4 shows an example of how the values are modified using [9].

Basically, total ordering is achieved for each pixel by considering more criterion (i.e., sum of pixel values in some predefined neighborhoods). Here, total ordering means that for all pixels, although some may assume the same intensity value, can be ranked and there will be no tie. The value of each pixel is then modified in the complete matching sequence (viz., the bin frequencies of H_0 and H_1) induced by this total ordering. Note that the number of pixels in H_0 and H_{w0} must be the same, i.e.,

$$\sum_{i=0}^{2^\tau-1} h_0[i] = \sum_{i=0}^{2^\tau-1} h_{w0}[i]. \tag{3}$$

When there is a mismatch, the histogram H_0 is scaled by the factor λ, i.e., $H_0'[i] \leftarrow H_0[i] \times \lambda$, where

$$\lambda = \Big(\sum_{i=0}^{2^\tau-1} H_{w0}[i]\Big) \div \Big(\sum_{i=0}^{2^\tau-1} H_0[i]\Big). \tag{4}$$

Due to rounding error, the remaining numbers, if any left, will be taken care by the lower bins in the histogram. The modified values, after applying exact histogram specification, become $P_L'(x,y)$. The final image $A^{e,w}$, which is encrypted and containing data, is formed by concatenation, i.e.,

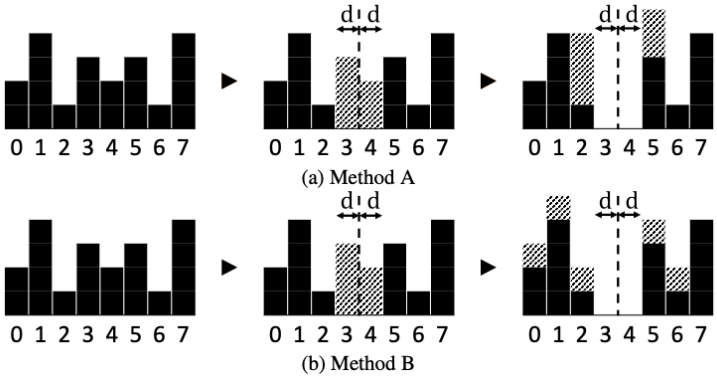


Fig. 5. Illustration of safety zone for $\tau = 3$.

$$A^{e,w}(x,y) = P'_M(x,y) \parallel P'_L(x,y). \quad (5)$$

In fact, based on the equation above, the proposed method is also *commutative* since P'_M and P'_L are completely independent. Specifically, *commutative* refers to the property where the exact same output can be obtained irregardless if the image is encrypted first followed by data insertion, or data is inserted first followed by encryption.

2.4 Decryption and Data Extraction

The proposed method is *separable* in which the decryption and extraction processes can take place independently. In other words, decryption can take place without remapping the pixel values to the corresponding (approximate) original values (i.e., reversing the data insertion process), and the data extraction process can take place independently without needing to first decrypt the image.

In the decryption process, the original location of the pixels in shuffled sub-image P'_M needs to be first regenerated using κ_2. Then, each P'_M is moved from their scrambled position (x', y') to the respective original location (x, y). Next, $P'_M(x, y)$ is XOR-ed with the keystream generated using κ_1 to obtain $P''_M(x, y)$. Then, the combination of the decrypted $P''_M(x, y)$ and $P'_L(x', y')$ produces the decrypted image $A^{d,w}$, which still contains the inserted data.

On the other hand, the inserted data can be extracted from the sub-image P'_L of the encrypted image $A^{e,w}$ as well as the decrypted image $A^{d,w}$. First, the sub-image $\{P'_L(x', y')\}$ is reshuffled using κ_3, where the pixels are reverted to their original positions (x, y). In addition, the parameter value τ and ϕ are required here to determine whether the value of $P'_L(x, y)$ encodes '0' or '1'. Specifically, when a value $P'_L(x, y)$ is included in H_0, then the extracted data at position (x, y) is $W'(x, y) = 0$. Otherwise $W'(x, y) = 1$. Note that H_0 and H_1 can be determined when τ and ϕ are known. The extracted data W' is obtained when all locations (x, y) are processed.

2.5 Post-processing

After extracting data W', $A^{d,w}$ can be further processed to obtain a better approximation of the original image A. Recall that during data insertion, the dynamic range of H'_0 and H'_1 are shrunk, and the values underwent a quantization-like process. Therefore, their original dynamic ranges and precisions need to be restored. Recall that the combined histogram of P''_M(after decryption) and P'_L resembles the histogram of the original image A. For that, the shape of H'_0 is modified by using Coltuc et al.'s method [9] to match that of A using P''_M and P'_L. The same operation is repeated for H'_1. The new image A^* is then generated by combining this new P_L and the decrypted P''_M.

3.0 IMPROVEMENT OF ROBUSTNESS

When the processed image (encrypted or inserted with data) undergoes any lossy compression or filtering operations, the process deteriorates the quality of the inserted data. As a result, the inserted data might lead to failure of the intended use such as authentication, ownership dispute, etc. Therefore, two techniques are proposed to make the inserted data more robust. For presentation purposes, $\varphi = 2$ is considered in this section.

(a) Lenna (b) Watermark for Lenna (c) Watermark for UCID images

Fig. 6: The Lenna test image and watermarks.

3.1 Safety Zone

When a pixel value is changed due to lossy compression / low-pass-filtering, the pixel value may 'spill' over the boundaries, i.e., modified to be similar with the neighboring values. This is the reason the quality of the inserted data deteriorates and it is difficult to recover completely. To address this issue, the concept of safety zone is introduced. Specifically, safety zone is a range of bin values in the histogram where the frequency of pixels within the said range are zero, i.e., empty. When a pixel value is changed due to lossy image compression, it is unlikely that it will spill over the boundaries due to the introduced safety zone. Let d be the size of safety zone for $0 \leq d \leq 2^{\tau}/2$, and two implementations of safety zone are shown in Fig. 5 for $\tau = 3$. Specifically, two selected bins, namely bin 3 and bin 4 in both examples, are vacated by means of distributing their elements to other bins. Here, Method A restricts the distribution process to the immediate adjacent bins, while Method B uniformly distributes the elements to all other non-safety-zone bins. Note that in both cases the histogram is empty for bin 3 and bin 4.

3.2 Majority Voting

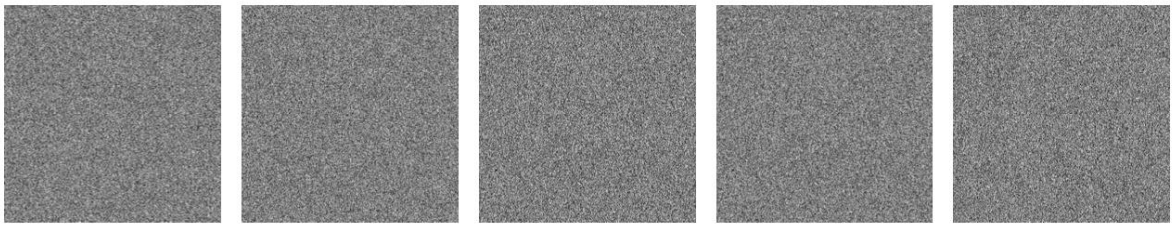
To further suppress the bit error rate of the extracted data after lossy compression or low-past-filtering, the payload size is reduced to 25% and so the same data (i.e., binary image W') can be repeatedly inserted into image A for 4 times. In the data extraction process, after four sets of data are obtained, the final hidden payload W' is considered by means of majority vote to reconstruct the inserted data. If there is a tie (i.e., 2 vs 2), a result is taken over a 3×3 neighborhood around the undecided pixel, and majority vote is deployed again. If there is undecided pixel, a random value (i.e., either '0' or '1') is output as the hidden payload bit.

4.0 EXPERIMENTS

The proposed JEDI method is implemented in MATLAB (Version R2017b - 9.3.0.713579). The test image Lenna (see Fig. 6(a)) and the Uncompressed Colour Image Dataset (UCID) [10] (converted to grayscale) are utilized to verify the performance of the proposed JEDI method. The UCID image dataset contains 1,338 images each of dimensions 512×384 (or 384×512) dimensions. It includes natural scenes and man-made objects, both indoor and outdoor images [10]. This dataset is carefully chosen due to its size, and it also caters to a wide variety of natural images, and hence the average results generated by using this dataset can be used to infer the general behaviour and performance of the proposed method. The binary images shown in Fig. 6(b) is inserted into the Lenna image and Fig. 6(c) is inserted into the UCID images throughout the evaluation process. For all experiments conducted, when using the correct key κ_3, it is verified that the extracted data is exactly the same as the original one. This happens irregardless if the data is extracted from $A^{e,w}$ and $A^{d,w}$. When the majority voting is adopted, the payload size is reduced to 25% of $M \times N$. Unless specified otherwise, the term *processed image* refers to an encrypted image with inserted data.

4.1 Perceptual Masking

As a representative example, a series of encrypted Lenna images containing inserted data are shown in Fig. 7. These images suggest that the proposed JEDI method is able to completely mask the perceptual semantic of the corresponding original image. That is, there is no trace of the original image. The corresponding PSNR and SSIM [11] values achieved by various combination of τ and φ are summarized in Table 1. Note that, PSNR ranges from [0, ∞] dB while SSIM ranges from [-1, 1]. Higher PSNR and SSIM values indicate better image quality (i.e., a measure of



(a) $\tau = 1$ (b) $\tau = 2$ (c) $\tau = 3$ (d) $\tau = 4$ (e) $\tau = 5$

Fig. 7: Encrypted Lenna image by using $\varphi = 2$.

Table 1: PSNR (dB) and SSIM for encrypted Lenna with different combinations of τ and φ.

ϕ	$\tau=1$		$\tau=2$		$\tau=3$		$\tau=4$		$\tau=5$		$\tau=6$		$\tau=7$	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
2	8.714	0.009	8.720	0.010	8.706	0.009	8.703	0.009	8.687	0.009	8.748	0.010	8.872	0.014
4			8.720	0.010	8.706	0.009	8.694	0.009	8.654	0.009	8.818	0.011	8.948	0.020
8					8.705	0.009	8.693	0.009	8.643	0.009	8.787	0.011	9.013	0.023
16							8.693	0.009	8.643	0.009	8.774	0.010	9.003	0.023
32									8.643	0.009	8.774	0.010	8.995	0.023
64											8.774	0.010	8.995	0.023
128													8.995	0.023

Table 2: Statistics of PSNR (dB) and SSIM for encrypted UCID images with different combinations of τ and φ.

ϕ		$\tau=1$		$\tau=2$		$\tau=3$		$\tau=4$		$\tau=5$		$\tau=6$		$\tau=7$	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
2	Ave.	7.351	0.007	7.350	0.007	7.347	0.007	7.351	0.008	7.376	0.008	7.465	0.010	7.869	0.020
	SD	0.788	0.002	0.787	0.002	0.788	0.002	0.785	0.002	0.768	0.001	0.722	0.003	0.711	0.011
4	Ave.			7.351	0.007	7.347	0.007	7.356	0.008	7.416	0.008	7.622	0.012	8.573	0.034
	SD			0.786	0.002	0.788	0.002	0.778	0.002	0.735	0.002	0.632	0.005	0.511	0.020
8	Ave.					7.348	0.007	7.357	0.008	7.422	0.008	7.673	0.012	8.825	0.039
	SD					0.787	0.002	0.778	0.002	0.728	0.002	0.592	0.006	0.320	0.023
16	Ave.							7.358	0.008	7.424	0.008	7.683	0.013	8.925	0.040
	SD							0.777	0.002	0.726	0.002	0.581	0.007	0.201	0.024
32	Ave.									7.425	0.009	7.688	0.013	8.953	0.041
	SD									0.726	0.002	0.577	0.007	0.152	0.025
64	Ave.											7.689	0.013	8.971	0.041
	SD											0.576	0.007	0.115	0.025
128	Ave.													8.975	0.042
	SD													0.099	0.025

similarity to the reference image), and vice versa. In general, the values are low for both PSNR and SSIM, hence suggesting that the perceptual semantic of the image is masked. The differences in results are also relatively small for all the considered parameters, hence suggesting the ability of the proposed JEDI method to mask the image irregardless of the parameter in use. To have a better understanding of the performance on natural images, the average results and standard deviation for the UCID images are recorded in Table 2. It is observed that UCID images show similar trend, i.e., the perceptual semantic of the images are masked which is suggested by the low PSNR (7.350 dB to 8.975 dB) and SSIM (0.002 to 0.042) values irregardless of τ and φ. The results are also consistent for all UCID images because the standard deviation for both PSNR (i.e., < 0.8dB) and SSIM (i.e., < 0.03) are small.

(a) $\phi = 2$ (29.00, 0.720) (b) $\phi = 4$ (34.86, 0891) (c) $\phi = 8$ (40.66, 0.966) (d) $\phi = 16$ (46.52, 0.991) (e) $\phi = 32$ (50.89, 0.997)

Fig. 8: Decrypted Lenna images using $\tau = 5$ without post-processing. The image quality is reported in the form of (PSNR, SSIM).

Table 3: PSNR (dB) and SSIM for decrypted Lenna with different combinations of τ and φ.

ϕ	$\tau=1$		$\tau=2$		$\tau=3$		$\tau=4$		$\tau=5$		$\tau=6$		$\tau=7$	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
2	51.137	0.996	46.356	0.987	40.724	0.956	34.779	0.857	28.076	0.612	23.371	0.386	16.900	0.153
4			51.106	0.996	46.358	0.987	40.675	0.955	33.126	0.811	28.991	0.661	22.759	0.366
8					51.066	0.996	46.279	0.987	39.658	0.945	34.029	0.843	31.347	0.761
16							50.806	0.995	44.977	0.983	41.113	0.960	36.345	0.897
32									49.511	0.994	46.104	0.987	41.563	0.964
64											50.765	0.995	46.153	0.987
128													50.620	0.995

Table 4: Statistics of PSNR (dB) and SSIM for decrypted UCID images with different combinations of τ and φ.

ϕ		$\tau=1$		$\tau=2$		$\tau=3$		$\tau=4$		$\tau=5$		$\tau=6$		$\tau=7$	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
2	Ave.	51.14	0.996	46.32	0.989	40.62	0.963	34.89	0.885	28.94	0.706	22.65	0.450	16.44	0.234
	SD	0.010	0.001	0.160	0.004	0.339	0.014	0.539	0.040	0.740	0.090	1.194	0.125	1.902	0.106
4	Ave.			50.98	0.996	46.08	0.988	40.81	0.964	35.22	0.891	28.70	0.706	22.25	0.466
	SD			0.371	0.001	0.653	0.005	1.068	0.016	1.574	0.055	2.433	0.129	3.454	0.173
8	Ave.					50.63	0.996	46.02	0.988	40.98	0.963	35.21	0.889	28.02	0.704
	SD					0.860	0.002	1.184	0.007	2.087	0.028	3.083	0.086	4.420	0.170
16	Ave.							50.33	0.995	45.89	0.986	40.53	0.955	34.25	0.875
	SD							1.535	0.004	2.209	0.017	3.626	0.056	5.224	0.124
32	Ave.									49.77	0.993	45.33	0.980	39.06	0.936
	SD									2.627	0.013	3.885	0.040	6.072	0.098
64	Ave.											48.68	0.988	44.15	0.968
	SD											4.228	0.034	6.481	0.075
128	Ave.													46.89	0.977
	SD													6.715	0.068

4.2 Decrypting the Processed Image

Each processed image is decrypted to generate an approximation of the original image $A^{d,w}$. Note that even after decryption, a perfect reconstruction is not possible because part of the image, viz., P_L', is reserved for data insertion where exact histogram specification [9] is performed. To investigate into the performance, the number of sub-histogram clusters φ is varied and the results are observed. Recall that P_L is utilized for histogram formation and distribution. Hence, the number of sub-histogram clusters φ is limited by the range of values (bins) in P_M, where $\varphi = 2^k, 0 \leq k \leq \tau$. The resulting decrypted images, $A^{d,w}$, using various φ are as shown in Fig. 8. It is observed that there is no visible distortion observed subjectively. We further analyze the images by measuring the PSNR and SSIM values of the decrypted UCID test images for various combinations of τ and φ. The average PSNR and SSIM are recorded in Table 3 for Lenna, and Table 4 for the UCID images. In general, the quality of the decrypted image is low when φ is small and τ is large, e.g., $(\varphi, \tau) = (2, 7)$. However, when φ increases while holding τ constant, the quality of the decrypted image improves (i.e., up to 50dB). It is because when φ increases, the dynamic range of H_0 and H_1 are widen for data insertion purposes. In other words, the values in H_0 and H_1 are mapped values that are closer to their

(a) $\phi = 2$ (49.34, 0.996) (b) $\phi = 4$ (49.57, 0.996) (c) $\phi = 8$ (49.94, 0.997) (d) $\phi = 16$ (50.38, 0.997) (e) $\phi = 32$ (50.75, 0.997)

Fig. 9: Decrypted Lenna images using $\tau = 5$ after post-processing. The image quality is reported in the form of (PSNR, SSIM).

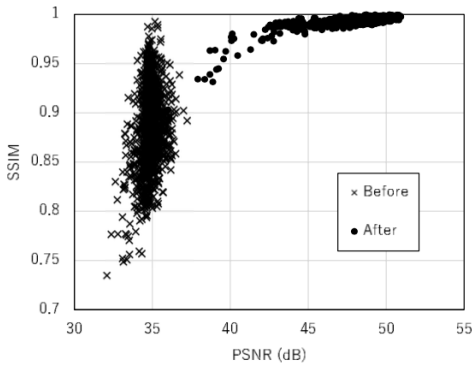


Fig. 10: Scatter plot of PSNR vs SSIM of before and after post processing for UCID images with $\tau = 4$ and $\varphi = 2$.

original counterparts (similar to fine quantization). In all cases, when both (φ, τ) are large, better quality is observed in the decrypted image. These outcomes also imply that the quality of the decrypted image can be controlled. In addition, part of the inserted data can be reserved for quality restoration purposes. These potential applications of the proposed JEDI method will be explored as our future work.

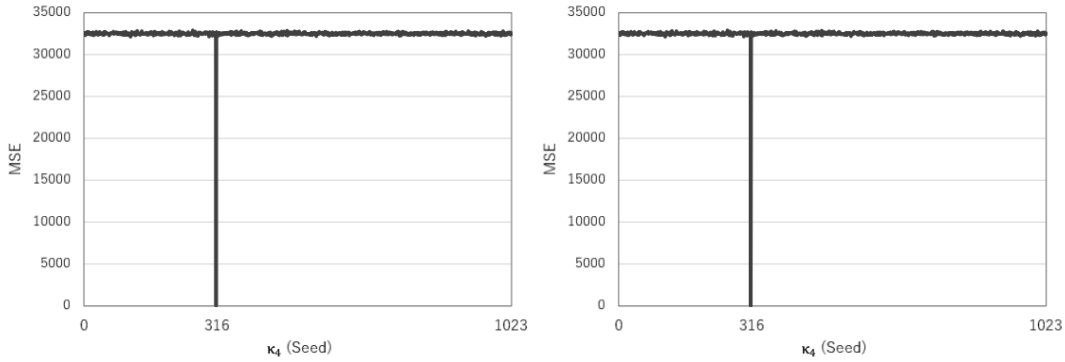
4.3 Post-processing

This section reports the results before and after applying the post-processing operations on the decrypted image $A^{d,w}$. As a representative example, Fig. 8 shows the decrypted Lenna images without post-processing, while Fig. 9 shows the corresponding images after performing post-processing. By visual inspection, it is apparent that the image quality improves after post-processing. This observation also agrees with the quantitative results. To further investigate into its effectiveness, post-processing operations are applied to the UCID images and the results are presented as a graph of SSIM vs PSNR in Fig. 10. It is observed that both SSIM and PSNR improve after performing post-processing, where the data points ('×') are shifted toward the upper-right corner (·). Therefore, the receiver has the option to improve the quality of the decrypted image by performing the post-processing operations detailed in Section 2.5.

4.4 Robustness against Unauthorized Viewing

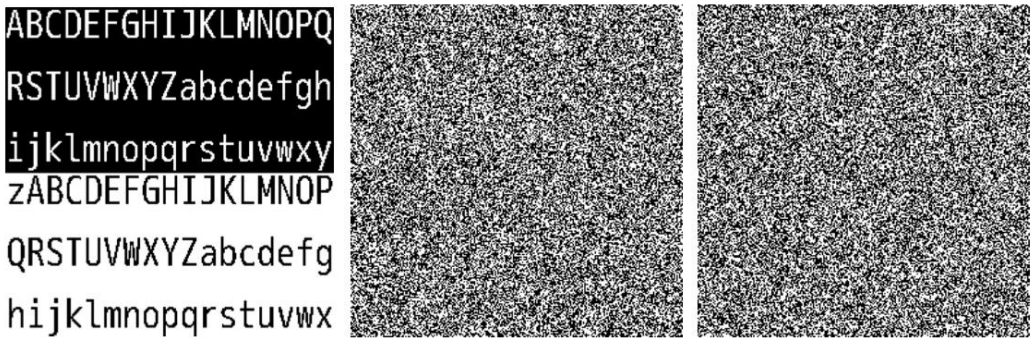
To verify the robustness of the inserted data with respect to unauthorized viewing, the data is inserted by using $\kappa_3 = 316$, but extracted by using incorrect keys κ_3'. The mean square error (MSE) between the original data W and the extracted one W' is computed for each of the 1024 keys considered, and the results are plotted in Fig. 11. It is obvious that only the correct key, i.e., κ_3'=316, is able to extract the inserted data correctly.

Similarly, we consider the decryption of the processed image. Again, decryption is attempted by using the wrong key κ_1' and κ_2'. When one of the keys are incorrect, it is found that output is completely gibberish. For example, Fig. 12(b) shows the output when κ_1 is incorrect, while Fig. 12(c) shows the output when κ_2 is incorrect. In both cases, the output image is nothing similar to the original image. Therefore, both keys are needed to decrypt the processed image correctly, as illustrated in Fig. 13(a).



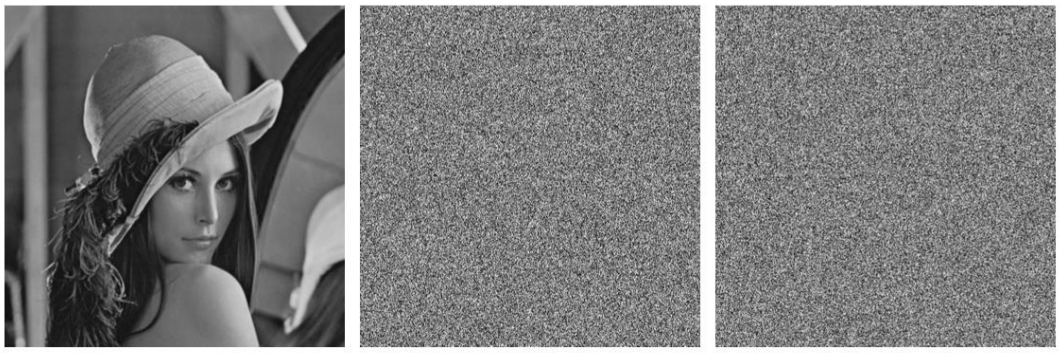
(a) From $A^{e,w}$ - encrypted image (b) From $A^{d,w}$ - decrypted image

Fig. 11: Graphs of MSE vs κ_3 (seed value) using $\tau = 3$ and $\varphi = 2$. Note that a smaller MSE implies better performance.



(a) Original (b) $\kappa_3 = 138$ (c) $\kappa_3 = 846$

Fig. 12: The binary image extracted from different images.



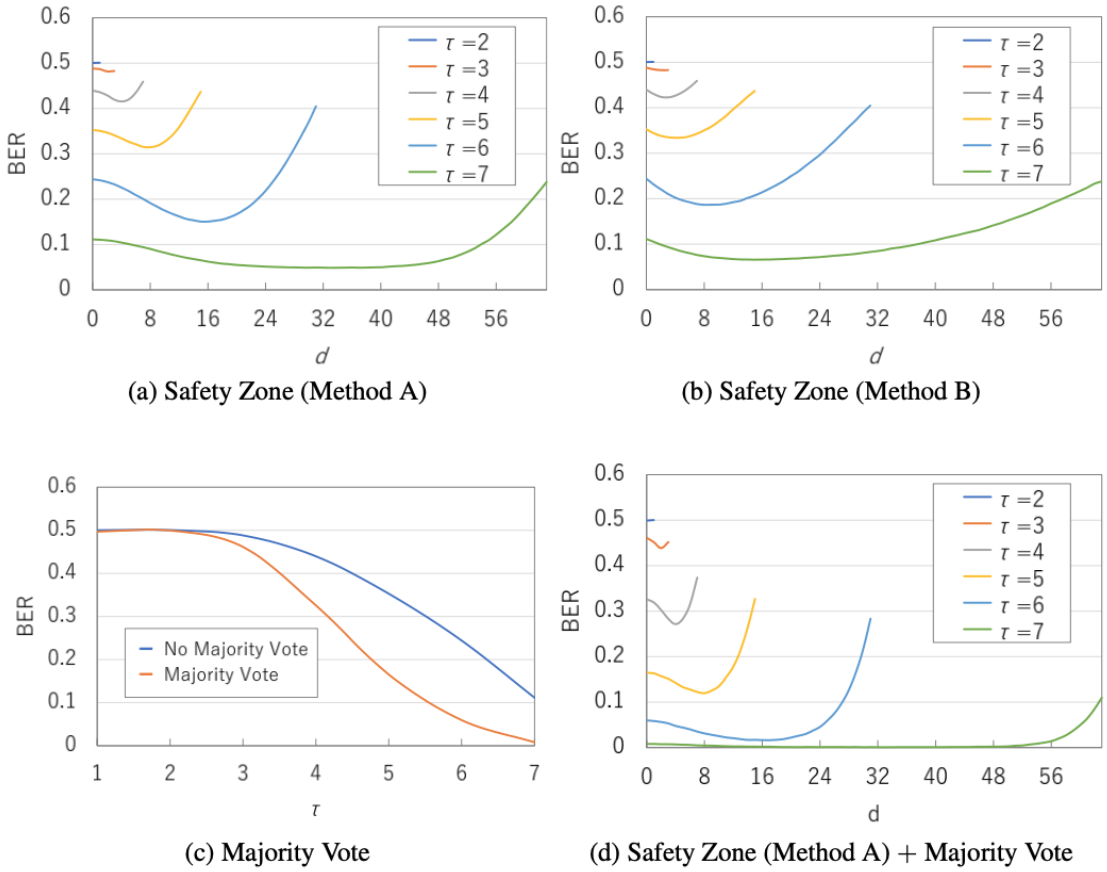
(a) $(1685, 3641)$ (b) $(842, 3641)$ (c) $(1685, 2419)$

Fig. 13: The decrypted images with different keys recorded in the form of (κ_1, κ_2).

4.5 Robustness against Compression and Image Filtering

To verify the performance of safety zone and majority vote, the processed image is compressed by using JPEG, and the data extraction procedure is applied on the resulting compressed image. Here, a quality factor of 75 is applied. The data extracted from the compressed image is compared with its original counterpart to compute the bit error rate (BER). Figure 14 shows the BER for different combinations of d and τ while setting $\varphi = 2$. Here, $d = 0$ refers to the scenario without using safety zone. Results suggest that the BER is reduced when either safety zone or majority vote is introduced. Figure 14(d) shows the results for the combined use of safety zone and majority vote. The combined approach yields an overall better results in comparison to the case when only one strategy is in use, which is an expected results. However, when majority vote is in use, the effective payload size is reduced, i.e., 25% of the original size.

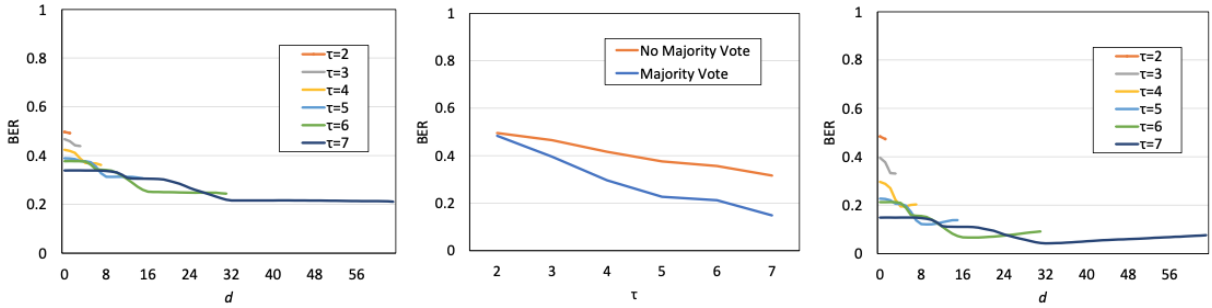
Next, low-pass filtering is performed on the processed image, and the data extraction procedure is applied. Here, the following 3×3 filter is used:



(a) Safety Zone (Method A) (b) Safety Zone (Method B)

(c) Majority Vote (d) Safety Zone (Method A) + Majority Vote

Fig. 14: BER for different d and τ value.



(a) Safety Zone (Method A) (b) Majority Vote (c) Safety Zone (Method A) + Majority Vote

Fig. 15: BER after low-pass filtering for different d and τ value.

$$\frac{1}{6}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}. \quad (6)$$

The BER results are shown in Fig. 15. When only one strategy is deployed, the BER is relatively high. Specifically, in the case of safety zone, BER is high (i.e., 0.5) when τ is small. While holding d constant, BER decreases when τ increases. BER further decreases when d increases. On the other hand, for the case of majority vote, although BER is lower after the strategy is deployed, BER is still high, i.e., 0.5 when $\tau = 2$. However, BER decreases to < 0.2 when τ increases to 7. When both strategies (i.e., safety zone and majority vote) are combined, BER can be effectively suppressed to < 0.1. The performance improves when τ increases because more bitplanes are allocated for data hiding purpose. Interestingly, for both JPEG compression and low-pass-filtering, the performance improves with d to a certain point, then deteriorates. A potential reason is that, when d increases initially, the values that were originally

Table 5: Functional comparison among the proposed and conventional JEDI methods.

Method	Scalability	Capacity	Reversible	Commutative	Separable	Robustness	Complexity
Zhang [4]	Yes	∼0.001 bpp	No	No	Yes	No	Low
Hong et al. [12]	Yes	∼0.016 bpp	No	No	Yes	No	Medium
Ong et al. [1]	Yes	∼2.88 bpp	Yes	No	No	No	Medium
Nguyen et al. [13]	No	<0.5 bpp	Yes	No	Yes	No	High
Li et al. [14]	No	1 bpp	Yes	No	Yes	No	Medium
Yi et al. [15]	Yes	$\leq$ 0.5 bpp	Yes	No	Yes	No	High
Qin et al. [16]	Yes	<0.05 bpp	Yes	No	Yes	No	High
Puteaux et al. [17]	No	∼1 bpp	Both	No	No	No	High
Wu et al. [18]	Yes	∼2.5 bpp	Yes	No	Yes	No	High
Wang et al. [19]	No	∼2.1 bpp	Yes	No	Yes	No	High
Yu et al. [20]	Yes	∼3.5 bpp	Yes	No	Yes	No	High
Proposed	Yes	1 bpp	No	Yes	Yes	Yes	Medium

within the safety zone are moved away from the boundary of H_0 and H_1. As a result, BER decreases. However, this gain diminishes when d is increased further because the values tend to cross the boundaries of H_0 and H_1. It is due to the fact that the bins in H_0 and H_1 are alternating (see Fig. 2). Therefore, there is a trade-off among BER, the effective payload size, as well as the quality of the decrypted image. However, in any case, based on the aforementioned results, safety zone and majority vote can complement the proposed JEDI method, making it more robust.

5.0 FUNCTIONAL COMPARISON

This section discusses the properties of the proposed JEDI method in comparison to several related state-of-the-art methods, including Zhang et al. [4], Hong et al. [12], Ong et al. [1], Nguyen et al. [13], Li et al. [14], Yi et al. [15], Qin et al. [16], Puteaux et al. [17], Wu et al.'s[18], Wang et al.'s [19], and Yu et al.'s [20]. Table 5 summarizes the functional comparison results for all considered methods from the aspects of scalability, capacity, reversibility, commutative, complexity, etc. Note that, the results are obtained by reviewing the methods carefully, and we refer the interested reader to their respective papers [4, 12, 1, 13, 14, 15, 16, 17, 18, 19, 20] for detailed explanations of the methods. All of these methods (including the proposed JEDI method) achieved sufficient perceptual masking quality while realizing data embedding. Besides, they are all JEDI methods operating in the spatial or pixel domains.

Zhang, Hong et al., Yi et al. and Qin et al.'s methods utilize certain block-based structure for the masking or/and embedding process. In these methods, only payload of limited size can be inserted into a single block. Hence, their embedding capacity is relatively lower. Furthermore, Yi et al., Qin et al., and Nguyen et al.'s methods adaptively insert data into selected blocks/pixels only by considering the block/pixel structures, viz., either smooth or complex. Hence, these methods also have limited embedding capacity despite being reversible. On the other hand, Li et al.'s method, Puteaux et al.'s method and the proposed method achieve relatively higher embedding capacity, which is 1 bpp, due to the pixel-based data insertion operations. In terms of masking, Li et al.'s method can flexibly employ any probabilistic cryptosystem, while Puteaux et al.'s method and the proposed method both perform XOR operation in concealing the perceptual semantics of the image. Recently. Wu et al., Wang et al. and Yu et al. also proposed to use an adaptive method in selecting embeddable pixels for high capacity embed- ding, while ensuring reversibility. These methods are able to hide more than 2 bpp of data excluding the side information needed for the decoding process. From Table 5, most methods achieve reversibility by sacrificing the embedding capacity, but the proposed method can achieve 1 bpp with near-reversible performance. Among the considered JEDI methods, Ong et al.'s method can achieve reversibility with the considerably high embedding capacity, because it combines the insertion and masking operations into one single step, where a modification to one pixel can lead to the encoding of multiple bits.

Next, the commutative and separable properties are discussed. Ong et al.'s method is non-commutative and non-separable because the data extraction and decryption are combined. Recent works [18, 19, 20] are able to achieve high

embedding capacity by adaptively identifying the embeddable locations for secret bit embedding. These methods are also separable. However, most compared methods in Table 5 are non-commutative due to the strict order of operations (in this case, always encryption first, then data hiding), except for the proposed method. It is because the proposed method separates the cover image into two non-overlapping parts, i.e., one for data insertion and the other for perceptual masking.

In terms of robustness, all the considered conventional methods do not include any strategies to withstand against compression or image filtering operation. By implementing safety zone and majority vote, the proposed method is able to withstand lossy compression (i.e., JPEG) and image filtering (i.e., low-pass-filtering).

Although the proposed method has additional features to enhance robustness, the computational complexity of the proposed method is similar to that of Hong et al.'s [12], Ong et al.'s [1] and Li et al.'s [14] methods. It is because these methods are using the global image features (e.g., image histogram). Among them, Zhang et al.'s method [4] has the lowest complexity since it only performs data hiding directly on the LSB of the image pixels, but its embedding capacity is also the lowest among all the considered methods. As for the other methods [13, 15, 16, 17, 18, 19, 20], they are of higher computational complexity than the proposed method because these methods are adaptive in nature, in which they need to walk through the image, pixel-by-pixel, to identify the suitable embedding strategies or locations, and most of them need to explicitly store extra information for perfect image reconstructions (i.e., reversibility).

All in all, the proposed JEDI method offers more features and robustness against JPEG compression and low-past-filtering. It also exhibits balanced performances, with near complete reversibility in addition to being commutative and separable.

6.0 CONCLUSION

In this work, a joint encryption and data insertion method is put forward. In particular, the input image is divided into two independent parts, where one part is manipulated to mask the perceptual semantic of the image, and another part is manipulated to insert data. The concept of safety zone is proposed and majority vote is adopted to improve robustness of the inserted data against common image processing operations. Experiment results suggest that, when compared with the conventional JEDI methods, the proposed method is more robust against JPEG compression and low-past-filtering. It is also commutative and separable.

As future work, the possibilities of inserting multiple images into a single image will be explored. In addition, we want to investigate into the relationship between the parameters (i.e., φ and τ) and the quality of the decrypted image, and how other post-processing operations can improve the quality of the decrypted image.

7.0 ACKNOWLEDGEMENT

This research is supported by the E-Science fund under the project - *Innovative High Dynamic Range Imaging - from Information Hiding to Its Applications* (Grant No. 01-02-10-SF0327).

REFERENCES

[1] Simying Ong, KokSheik Wong, and Kiyoshi Tanaka, "A scalable reversible data embedding method with progressive quality degradation functionality," *Signal Processing: Image Communication,* vol. 29, no. 1, pp. 135–149, 2014.

[2] Simying Ong, KokSheik Wong, and Kiyoshi Tanaka, "Scrambling embedding for jpeg compressed image," *Signal Processing,* vol. 109, pp. 38–53, 2015.

[3] Mustafa S. Abdul Karim and KokSheik Wong, "Data embedding in random domain," *Signal Processing,* vol. 108, pp. 56–68, 2015.

[4] X. Zhang, "Reversible data hiding in encrypted image," *IEEE Signal Processing Letters,* vol. 18, no. 4, pp. 255–258, April 2011.

[5] M. Cancellaro, F. Battisti, M. Carli, G. Boato, F. G. B. De Natale, and A. Neri, "A joint digital watermarking and encryption method," *Proceedings of SPIE 6819, Security, Forensics, Steganography, and Watermarking of Multimedia Content X, 68191C,* 2008.

[6] Jen-Chun Chang, Yi-Zhi Lu, and Hsin-Lung Wu, "A separable reversible data hiding scheme for encrypted jpeg bitstreams," *Signal Processing*, vol. 133, no. C, pp. 135–143, Apr. 2017.

[7] Dawen Xu, Rangding Wang, and Yun Q. Shi, "Data hiding in encrypted h.264/avc video streams by codeword substitution," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 4, pp. 596–606, Apr. 2014.

[8] Yiqi Tew, KokSheik Wong, Raphael C.-W. Phan, and King Ngi Ngan, "Separable authentication in encrypted hevc video," *Multimedia Tools and Applications*, pp. 24165–24184, Sep. 2018.

[9] D. Coltuc, P. Bolon, and J. M. Chassery, "Exact histogram specification," *IEEE Transactions on Image Processing*, vol. 15, no. 5, pp. 1143–1152, May 2006.

[10] G. Schaefer and M. Stich: "UCID - An Uncompressed Colour Image Database," *Proc. SPIE: Storage and Retrieval Methods and Applications for Multimedia*, vol. 5307, pp. 472–480 (2003)

[11] ZhouWang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

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

[13] T.-S. Nguyen, C.-C. Chang, W.-C. Chang, "High capacity reversible data hiding scheme for encrypted images," *Signal Processing: Image Communication*, vol. 44, pp. 84-91, 2016.

[14] M. Li, and Y. Li, "Histogram shifting in encrypted images with public key cryptosystem for reversible data hiding," *Signal Processing*, vol. 130, pp. 190-196, 2017.

[15] S. Yi, Y. Zhou, and Z. Hua, "Reversible data hiding in encrypted images using adaptive block-level prediction-error expansion," *Signal Processing: Image Communication*, vol. 64, pp. 78-88, 2018.

[16] C. Qin, W. Zhang, F. Cao, X. Zhang, and C.-C. Chang, "Separable reversible data hiding in encrypted images via adaptive embedding strategy with block selection," *Signal Processing*, vol. 153, pp. 109-122, 2018.

[17] P. Puteaux and W. Puech, "An Efficient MSB Prediction-Based Method for High-Capacity Reversible Data Hiding in Encrypted Images," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 7, pp. 1670-1681, July 2018.

[18] Y. Wu, Y. Xiang, Y. Guo, J. Tang and Z. Yin, "An Improved Reversible Data Hiding in Encrypted Images Using Parametric Binary Tree Labeling," *IEEE Transactions on Multimedia*, vol. 22, no. 8, pp. 1929-1938, Aug. 2020.

[19] Y. Wang and W. He,"High Capacity Reversible Data Hiding in Encrypted Image Based on Adaptive MSB Prediction," *IEEE Transactions on Multimedia*, March 2021.

[20] C. Yu, X. Zhang, X. Zhang, G. Li and Z. Tang, "Reversible Data Hiding with Hierarchical Embedding for Encrypted Images," *IEEE Transactions on Circuits and Systems for Video Technology*, March 2021.