Multimed Tools Appl
DOI 10.1007/s11042-015-2917-6

Encryption domain content-based image retrieval and convolution through a block-based transformation algorithm

Jia-Kai Chou[1] · Chuan-Kai Yang[1] · Hsing-Ching Chang[2]

Received: 28 January 2015 / Revised: 6 August 2015 / Accepted: 24 August 2015

Abstract In this paper, we propose a block-based transformation algorithm to achieve the purpose of image content protection. More importantly, under the proposed image content protection framework, image retrieval and image convolution can also be performed directly on the content-protected images. As a consequence, not only secure image storage and communication are accomplished, but also the computation efforts can be fully distributed, thus making it a perfect match for nowadays popular cloud computing technology. Security analyses are conducted to prove that the proposed image encryption scheme offers certain degree of security in both statistical and computational aspects. Although a higher data confidentiality may be reached by adopting traditional cryptographic encryption algorithms, we believe it could be accepted by ordinary users with general image storage needs, since extra functionalities, i.e. content-based image retrieval and image convolution, are provided. Experimental results also demonstrate the decent performance of the proposed encryption domain image retrieval and convolution with acceptable storage overhead.

Keywords Image transformation/encryption · Ecryption domain content-based image retrieval/convolution

✉ Jia-Kai Chou
D9809101@mail.ntust.edu.tw

Chuan-Kai Yang
ckyang@cs.ntust.edu.tw

Hsing-Ching Chang
D9309104@mail.ntust.edu.tw

[1] National Taiwan University of Science and Technology, Taipei City, Taiwan, Republic of China

[2] Chungyu Institute of Technology, Keelung City, Taiwan, Republic of China

Published online: 24 September 2015

1 Introduction

Thanks to the rapid evolution and fast development of emerging cloud computing technology, individuals as well as companies transfer huge amounts of digital data onto remote servers, so that maximum mobility and flexibility of data accessing and processing can be achieved. However, the privacy of the deposited data can be very vulnerable if the data are stored in their plain form. Once a malicious attacker cracks into the system, the contents of the data may be fully revealed; or even worse, if an untrustworthy server itself acts as a data theft, it may steal the private information from users without any extra effort. A potential application scenario would be for nowadays popular cloud image storage services, like Google Photo, Flickr, Instagram, etc, which provide reliable online image backup/storage service with acceptable fees or even for free. Users upload huge amounts of their daily life images everyday. Although the companies usually claim that the images are safely deposited in their servers and will not be analyzed or touched for their internal use, there is still no privacy guarantee if anybody gets unauthorized access to the images because the images are stored in their plain form. As a consequence, maintaining the privacy over the remotely stored data is a serious issue to be considered under the client-server data storage framework.

Due to the fast growth of the consumer electronics industry, hundreds of thousands of digital images are taken and uploaded to the web everyday. An on-line image storage service with confidentiality would be really desirable for the users who want their images kept secret while transmitting over the Internet. In a remote image database system with an enormous amount of image data, searching and indexing of the stored images would be helpful for the users to retrieve desired relevant images efficiently. Thus, efficient image encryption/decryption techniques as well as effective image retrieval methods would play critical roles in such an image exchange model.

Image convolution is a basic operation and is commonly applied on images to accomplish particular purposes. For example: image blurring, image sharpening, edge detection, etc. As a result, these two core functionalities, i.e. content-based image retrieval and image convolution, are usually regarded as the fundamental components in a client-server-based image storage service. It would be even nicer, if we could make use of the cloud computing technology/environment to delegate these two operations to the server side, so that the computational overhead on the client side can be further alleviated.

In this paper, we propose a block-based transformation algorithm through which images can be kept secret from unauthorized viewers. The main contribution of this paper is the additional functionalities of content-based image retrieval and convolution that can be carried out in the encryption domain without revealing the contents of the images. Security analyses are conducted to evaluate the degree of security that can be reached by the proposed method in both statistical and computational aspects. Moreover, the performance of image encryption/decryption/retrieval/convolution is measured by time, storage overhead, precision rate, image quality, and induced errors, etc.

The rest of the paper is organized as follows. Section 2 reviews literature that are closely related to this research. Section 3 describes the main image transformation scheme used throughout the proposed system. Section 4 illustrates how content-based image retrieval and convolutional image processing can be done in the encryption domain. Section 5 demonstrates our results and performs a thorough evaluation in terms of many different perspectives to prove the efficiency and effectiveness of the proposed framework. Section 6 concludes our work and hints at directions for potential further studies.

2 Related work

Image encryption as well as image retrieval and convolution in the encryption domain are the main fields that are directly related to this work.

2.1 Image encryption

There are generally two types of previously proposed image encryption mechanisms: homomorphic image encryption and image encryption using block-based transformation.

Several studies [11, 26, 28] have reviewed the concepts of homomorphic encryption techniques. However, the computation efficiency still remains a major issue for the homomorphic encryption algorithms to be practical in real use. In addition, due to the fact that the data sizes of images are typically larger than textual data, some of the (somewhat) homomorphic encryption algorithms that are appropriate for textual data may not be as efficient and suitable for images. Recently, some researchers has adopted *Shamir's Secret Sharing (SSS)* [32] for the purpose of processing multimedia data, especially images, in the encryption domain. Under the scheme, additive and multiplicative homomorphic properties can be carried out on the obfuscated data. This kind of approach is particularly applicable in a secure multi-party computation (SMC) [39] environment, where specific tasks are achieved by different parties while keeping everyone oblivious to each other. Nevertheless, the main downside to such an approach is the presumption that different parties/servers are not able to retrieve the obfuscated data from each other. That is, if some (or all) of the parties/servers are with malicious intentions, the privacy of delegated data will then become very vulnerable.

Block-based transformation algorithms typically decompose the input images into equal-sized blocks, and then each block is transformed or encrypted separately, through various techniques. Mitra et al. [27] applied random bit, pixel and block permutation on a single image to make the encrypted image look like a noisy one. Younes et al. [41] adopted a similar approach to achieve the goal of image encryption. Maniccam et al. [24] performed both lossless image compression and encryption by recording the basic scan patterns in the decomposed image sub-regions. Yoon et al. [40] permuted image pixels according to a pre-generated permutation matrix based on chaotic maps. Several works [12, 17, 31] encrypted input images by applying XOR operations, and this kind of image encryption scheme can also be generalized to videos very easily. Zeng et al. [42] encrypted images in the frequency domain instead of the spatial domain, and then further extended the encryption scheme from images to videos.

2.2 Image retrieval and convolution in the encryption domain

Image retrieval has been a well-studied research field over the past few decades. However, rare works have devoted their efforts to achieve the functionality of image retrieval in the encryption domain. It is because extracting the image features, such as color histogram, color moment, etc., is not straightforward as the image pixel values are changed after encryption. Zhang and Cheng [43] proposed to encrypt JPEG images by permuting the DCT coefficients while still allowing histogram-base image retrievals. Lu et al. [22, 23] proposed a feature-based image retrieval system by encrypting an image itself and its features entirely separately. Both works mainly focus on the problem of image feature protection that allows the computation of similarity measures to be performed in the encryption domain, so that secure content-based image retrieval *(CBIR)* can be achieved. However, the image

content itself is only assumed to be secure enough against attacks through other encryption methods.

Erkin et al. [8] surveyed the recent techniques on retrieving and processing multimedia contents based on homomorphic encryption, and they have noted that some operations, such as addition and multiplication, could be carried out in the encryption domain under certain constraints. More recently, Gentry [13] proposed to compute arbitrary functions over the encrypted data. Nonetheless, as mentioned above, traditional cryptographic schemes may not be well-suited for multimedia files due to the concerns of efficiency. In addition, noises or errors may occur when complicated functions are adopted.

There has been some research dedicated towards performing certain computation/ processing over the encrypted data based on *Shamir's Secret Sharing*. Upmanyu et al. [36] enabled the change detection of image frames extracted from video stream so as to achieve the purpose of video surveillance. Lathey et al. [20] proposed an efficient and (possibly) error-free mechanism to apply low-pass filtering to the encrypted data. Lathey et al. [19] also proposed to perform several image enhancement operations to the encrypted image data. However, as addressed above, the studies adopting *Shamir's Secret Sharing* scheme to protect data privacy basically presumes the cloud service providers are trustworthy and sensitive information will not be exchanged among each other. In this paper, we argue that this assumption is not always true. Therefore, the privacy of the delegated data should be preserved by the encryption scheme itself, but not determined by the service providers.

As for image convolutions/retrieval with block-based transformation algorithms, there is little previous work regarding such issues. It is mainly because a block-based transformation algorithm typically applies non-linear mappings to the original image pixel values, e.g. XOR, thus destroying its original structure. As a result, it becomes possibly more difficult to perform the desired convolutional operations in its corresponding encryption domain. Chu et al. [5] managed to conquer the difficulty of comparing pixel values in the encryption domain so as to achieve the goal of detecting moving objects in a surveillance video stream. However, in [5], the private key used for encrypting the image frames in one surveillance period should stay the same for such a scheme to work, thus reducing the attacker's effort for breaking the system. In the proposed method, the content-based image retrieval can still be done when different private keys are assigned to different images. The proposed encryption domain image retrieval approach shares a similar concept with Chu et al.'s work [5], as additionally the functionality of encryption domain image convolution is also made available. As a result, our work can be extended to various types of applications, e.g. image blurring, image sharpening, edge detection, etc, where the operations can be viewed as different types of image convolutions.

3 Image encryption scheme

This section presents how an input image is processed to reach the goal of image content protection. As the proposed method applies block-based transformation algorithm to reach the goal of image encryption, the terms "encrypt" and "transform" will then be used interchangeably hereafter.

3.1 Overview

We first start by describing the workflow of the proposed image encryption/decryption scheme as shown in Fig. 1. The step-by-step explanation is given as follows. First, a user

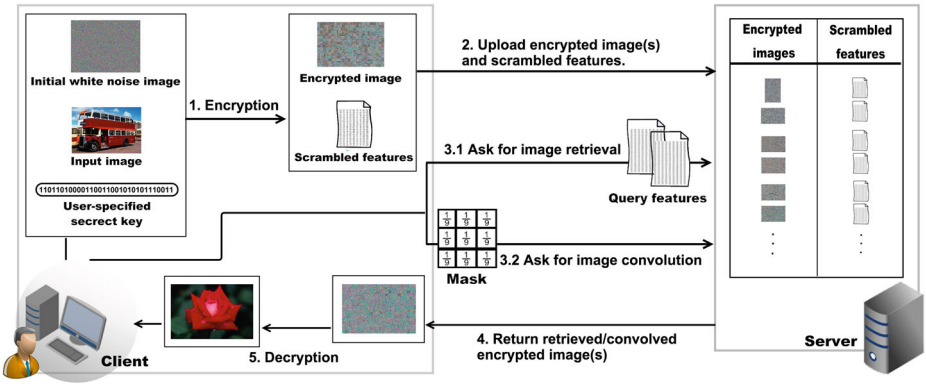


Fig. 1 The workflow of the proposed image encryption/decryption scheme

obtains an (or some) input image(s) that has privacy concern and needs to be protected. With some initial setup, such as creating a white noise and assigning secret key(s), the system then applies the proposed block-based transformation algorithm to encrypt the input image(s) and computes some scrambled image features for future image retrieval purpose. Second, the user uploads/stores the encrypted image(s) and the scrambled image features to the server side. Third, the user can request one of the two functionalities, i.e. image retrieval or image convolution, to be performed while sending out some required information to the server side. Fourth, the server side then processes its stored encrypted images according the user's request. Finally, the user receives the returned encrypted image(s) from the server side and executes the decryption to get the final image result.

3.2 Block-based image transformation

To support convolutional operations on encrypted images while achieving the functionality of image retrieval at the same time, the efficiency issue could be a major concern if we resort to traditional encryption schemes. One could use traditional encryption methods, e.g. Advanced Encryption Standard or homomorphic encryption, for achieving an extremely high level of security. However, it poses further challenges of processing the data in the encryption domain in terms of both storage and computation time. As a result, we intend to propose a more efficient image encryption framework to accomplish our goals. We opt to use a block-based transformation algorithm.

Like some of image encryption approaches that adopt block-based transformation algorithms, we first partition an input image, denoted as I_a and shown in Fig. 2a, into non-overlapping blocks. Next, the orders of the blocks are randomly permuted to perturb the content in the original image. Figure 2b shows an example of the permuted input image, denoted as P_a. Some well-known pseudo random number/index generators, such as linear feedback shift register *(LFSR)* [38] and Fisher-Yates shuffling [10], are usually used to generate the permutation sequences of the blocks.

As stated in [6], Demaine et al. have proved that, while solving a puzzle-like problem, if two pieces should be put together or not can only be answered after the entire puzzle is solved, then such a problem can be shown to be a NP-complete problem. That is, for both a human and a computer, exhaustively search for all possible permutations is necessary. Therefore, increasing the difficulty of matching two originally adjacent blocks can be

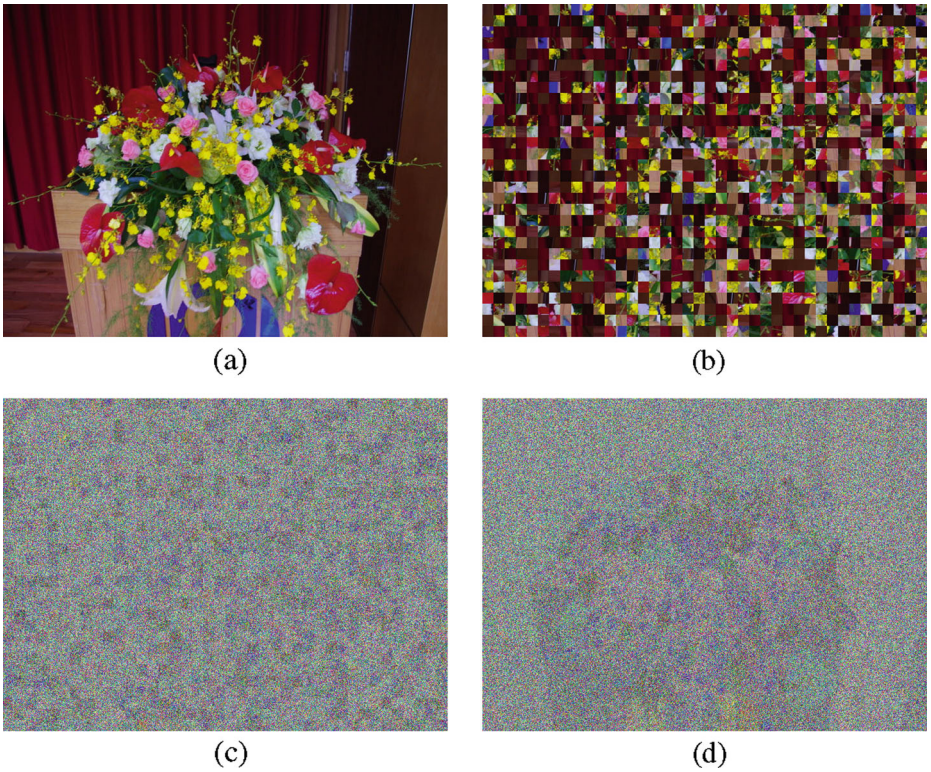


Fig. 2 **a** Original image that needs to be secured, denoted as I_a. **b** Permuted I_a, denoted as P_a. **c** Result of the encrypted I_a using the proposed method. **d** Compositing the original image I_a with white noise images and its complementary image without permutation. The shapes of some objects are somewhat recognizable

regarded as enhancing the secureness of the image encryption scheme. In order to achieve that, we propose to composite the permuted original image with white noise images so that the content in the original image could be protected:

$$\begin{aligned} M_1 &= w_1 \cdot P_a + w_2 \cdot N_1 \\ M_2 &= w_3 \cdot (255 - P_a) + w_4 \cdot N_2 \\ E_a &= w_5 \cdot M_1 + w_6 \cdot M_2 \end{aligned} \tag{1}$$

where E_a refers to the final transformed result of I_a, M_1 and M_2 are the intermediate images used to yield the final result, and N_1 and N_2 are two different white noise images. Moreover, each pair of the w_is is the weighting used to linearly combine two images into one and their values are added to one. The complement of the permuted image, $(255 - P_a)$ in the equation, has the effect of neutralizing the significant features in the original image, so as to make the resulting encrypted image more obscured. As can be seen in Fig. 2c, the resulting encrypted image looks meaningless. Note that the permutation during such an image encryption scheme is critical. As the examples demonstrated in Fig. 2d, when the permutation is ignored, the content and features of the original image could still be somewhat observed. But if permutation is involved in the encryption process, recovering the non-permuted version would cost a huge amount of computation. In addition, because the

target answer to this puzzle, Fig. 2d, is also posed in a vague form, the required complexity to reverse the encryption process can only be much higher.

However, such a framework may lead to security leakage if N_1 and N_2 are used constantly. That is, if the same pair of the white noise images is applied to transform every input image that needs to be protected, the approximated appearances of the images could be compromised by simply computing the differences between any two encrypted images. As another encrypted example provided in Fig. 3a, denoted by E_b, which is composited using the same pair of white noise images as in Fig. 2c, denoted by E_a. If we calculate the differences of pixel values between E_a and E_b, and plus 128 to each pixel for displaying purpose, some content of the image is then revealed in the differential image as shown in Fig. 3b. Therefore, as a final adjustment, we need to make sure that the white noise images used to encrypt different images do not stay identical. Figure 3c presents another encrypted result of I_a using a different pair of white noise images, and Fig. 3d shows the differential image between (a) and (c). Evidently, the content and the features of the images can be kept confidential by adopting distinct pairs of white noise images during each encryption process.

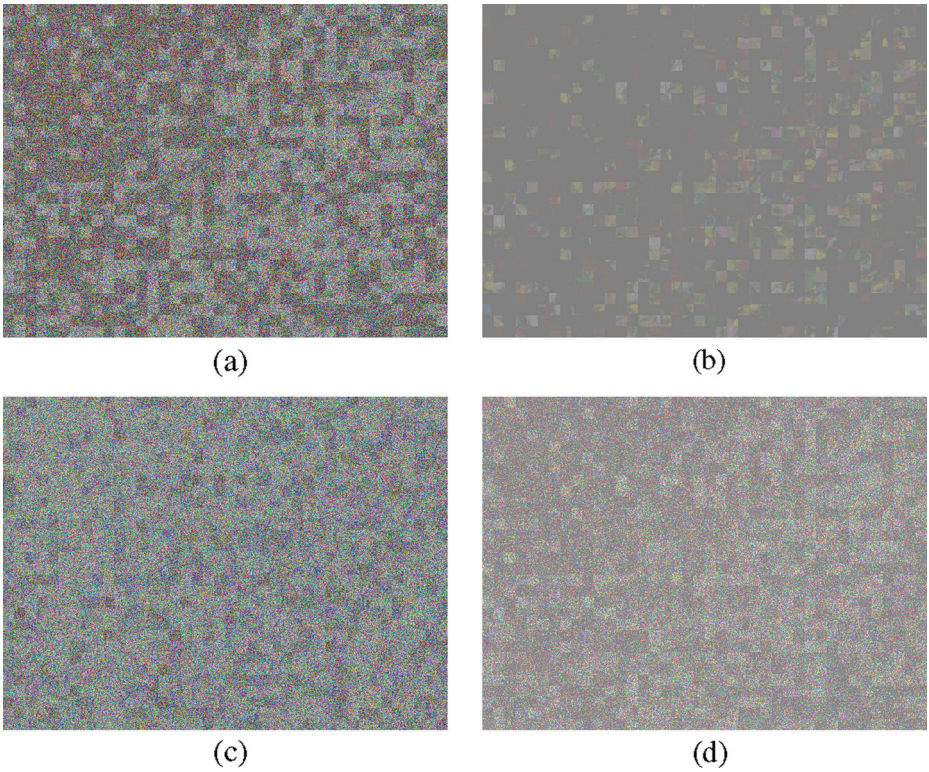


Fig. 3 **a** The encrypted result of a different input image, denoted by E_b, using the same white noise images as in Fig. 2c, denoted by E_a. **b** The differential image between the two encrypted images, E_a and E_b, using the same pair of white noise images, and the content is somehow revealed. **c** Another result of encrypted I_a, but with a distinct pair of white noise images. **d** The differential image between the two encrypted images where different white noise images are used. The content and features can be kept more secretly

3.3 Encryption key configuration

The proposed image transformation scheme can be regarded as a symmetric-key encryption scheme in traditional cryptographic encryption, which requires the same secret key in both encryption and decryption. The parameters used for transforming the image are regarded as secret keys and should be kept by the user. Only authorized users with the correct parameters are able to inversely transform the original image properly. On the other hand, it can be shown that, an unauthorized user, even with access to the transformed images, would fail to uncover the information of the original images. In the proposed approach, to obtain the original image content, only basic mathematic calculations are needed. This can be seen by first combining all formulas in (1):

$$\begin{aligned} E_a = {} & (w_5 \cdot w_1 - w_6 \cdot w_3) \cdot P_a + (w_6 \cdot w_3 \cdot 255) \\ & + (w_5 \cdot w_2 \cdot N_1) + (w_6 \cdot w_4 \cdot N_2) \end{aligned} \tag{2}$$

so the permutation of the original image is derived by:

$$P_a = \frac{E_a - w_6 \cdot w_3 \cdot 255 - (w_5 \cdot w_2 \cdot N_1) - (w_6 \cdot w_4 \cdot N_2)}{w_5 \cdot w_1 - w_6 \cdot w_3} \tag{3}$$

In order to get the correctly decrypted outcome, one has to know the weightings and the white noise images used in its encryption, which are w_1 to w_6, N_1 and N_2 in (3), as well as the permutation sequences. As a result, these parameters are treated as the *secret keys* in the proposed image encryption scheme.

As mentioned, the white noise images, N_1 and N_2, adopted during each transformation process need to be different for a higher degree of security. We propose a simple and efficient algorithm to meet this requirement with only one single user-specified key and one single constant white noise image, N. The encryption key configuration during the entire transformation process is described as follows.

1. An initial white noise image, denoted by N, is loaded and a user-specified key is given, where the size of the white noise image can vary and the key can be set to 128 bits, or even longer, depending on the user's desired security level. Note that N can be acquired from either a true random white noise or by adopting a cryptographically secure pseudo-random number generator, such as *Yarrow* [18] or *Fortuna* [9].
2. The user-specified key is then fed into a pseudo random number/index generator to generate the parameters needed in the ensuing image encryption process. The parameters, w_1 to w_6 and the permutation sequences, can be directly derived through the generator. Moreover, two newly synthesized, non-repeated white noise images, N_1 and N_2, are assembled by randomly selecting image patches from N. Figure 4 illustrates a partial example of the random selection procedure. The pseudo random number generator outputs random positions, and because the sets of randomly selected positions do not repeat in any two image transformation processes, one constant N can then be used to transform different input images.
3. As a final note, the values of w_1 and w_3 should be further controlled to ensure two things. One is that w_1 and w_3 have to be set to be relatively smaller than w_2 and w_4, so that the content of the original image does not predominate in the transformed image and most of the details can be perturbed by the white noise images. In our implementation, we limit the values of w_1 and w_3 to be smaller than 0.3. The other one is to assure that the value of $(w_1 \cdot w_5 - w_3 \cdot w_6)$ does not become zero, or the inverse transformation process would fail because of the exception of division by zero.

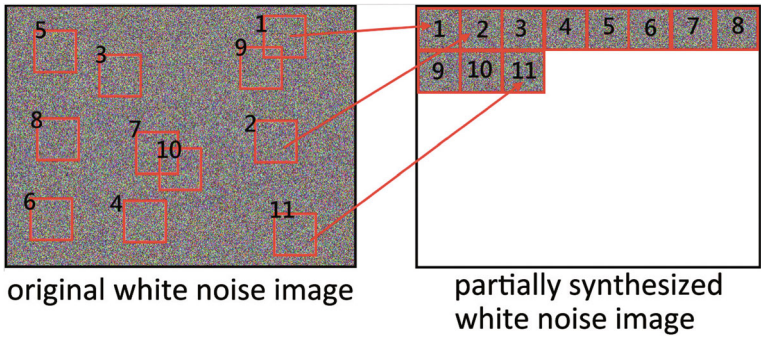


Fig. 4 The *left* image refers to the initially input constant white noise image, N. After the random positions being generated by the pseudo random number generator, the corresponding patches in the original white noise image are then selected to synthesize the new white noise image from *left to right*, and *top to bottom*. The right image shows a partially synthesized result during the process

Some may find the proposed image encryption scheme to be similar with the classical one-time pad method. In fact, in the proposed approach only the user-given 128-bit key will be changing every time when a new image comes in and needs to be encrypted. The 128-bit key is fed into a pseudo random number generator to yield the weighting parameters and choose patches from the initial white noise image, which simulates a similar effect as the one-time pad scheme does. The initial white noise image, N, is used repeatedly and the size of N does not incrementally affect the key size when more images are encrypted.

4 Encryption domain content-based image retrieval and image convolution

This section describes how the functionalities of content-based image retrieval and image convolution can be reached in the proposed encryption domain.

4.1 Encryption domain content-based image retrieval

To achieve the goal of encryption domain image retrieval, Lu et al. [22, 23] proposed to encrypt images and their associated features separately, where complex encryption schemes are involved to approximately maintain the ordering of the relevance among image features. However, we argue that it is not necessary to secure the image features using complicated encryption techniques, since supporting the functionality of image retrieval will inevitably disclose some information regarding the similarities among images. To effectively provide such functionality, the primary principal is to conceal the real values of the image features but keep the differences among the image features still comparable. A less complicated and more efficient approach would be more favorable.

A much simpler image feature protection mechanism is proposed. First, the initially user-given constant white noise image, N, is used as a base image. Moreover, the image features of N are computed as the base image features. Then, for every input image, before its transformation, we calculate the distances between its image features and the base image features. Finally, the calculated distances are assigned to be the transformed image features of the original input image. For example, assume that there are 3 image

features to be extracted from an image, so the base image features can be represented as $im\_f_{base} = [f_{base,1}, f_{base,2}, f_{base,3}]$. If we wish to compare two images, say A and B, while their image features are represented as $im\_f_a = [f_{a,1}, f_{a,2}, f_{a,3}]$ and $im\_f_b = [f_{b,1}, f_{b,2}, f_{b,3}]$, respectively. According to the aforementioned image feature transformation scheme, their corresponding transformed image features are then computed as $trans\_im\_f_a = [f_{a,1} - f_{base,1}, f_{a,2} - f_{base,2}, f_{a,3} - f_{base,3}]$ and $trans\_im\_f_b = [f_{b,1} - f_{base,1}, f_{b,2} - f_{base,2}, f_{b,3} - f_{base,3}]$, respectively. Apparently, in this way the real values of the image features in the original images can be perturbed by the base image features. Meanwhile, to estimate the distances between any two images is equivalent to measuring the distances among their corresponding transformed image features. Therefore, when an encrypted input image is to be transmitted to the remote server, uploading its corresponding sets of the transformed image features together onto the remote server will allow the functionality of image retrieval to be executed at the server side.

The above method is able to achieve the exact image retrieval results using any kind of image features without compromising the privacy of the images. Nonetheless, the computation effort for obtaining the image features is fully taken by the user side. Under a client-server data storage structure, it is more favorable to reduce the computational load from the user side as much as possible. We propose another secure content-based image retrieval scheme based on color histogram where the calculations of the image color histograms and comparisons are totally done at the sever side.

In (2), a transformed image can be divided into 3 parts. Part $A = (w_1 \cdot w_5 - w_3 \cdot w_6) \cdot P_a$ is the part related to the original image. Part $B = w_3 \cdot w_6 \cdot 255$ is a fixed number for every transformed image. Part $C = w_2 \cdot w_5 \cdot N_1 + w_3 \cdot w_6 \cdot N_2$ is the part containing random noise values. Because part C is a combination of two random white noise images, its color histogram will remain the characteristic of a white noise image: random and evenly distributed. Merging part B with part C means to shift the histogram of part C horizontally according to the value of part B, which still yields a random and evenly distributed histogram. Then, consider the situation when part A is also involved. Let $W = (w_1 \cdot w_5 - w_3 \cdot w_6)$. The value of W acts as a key influential factor. If W is a positive number, then the pixel values of the original image tend to push the histogram of the transformed image rightwards. Otherwise, the histogram of the transformed image is pushed leftwards. Bigger pixel values push the histogram towards the associated direction harder while smaller ones cause less effect. Hence, for an original image with lots of bigger pixel values and another original image with lots of smaller pixel values, the color histograms of their corresponding transformed images will be different from each other. As a consequence, comparing the distance based on color histogram between any two original images can be done by comparing the color histograms in their transformed forms.

Figure 5 displays three examples of the R channel histograms before/after the proposed image transformation. The weightings for these examples are set as the following: $w_1 = 0.03, w_2 = 0.97, w_3 = 0.227, w_4 = 0.773, w_5 = 0.07$ and $w_6 = 0.93$. Thus, $W \cong -0.209$, which means the pixel values of the original images will push the resulting histogram leftwards. Figure 5a and d show three original images and their associated transformed results, respectively. Figure 5b presents the R channel histograms of the original images while Fig. 5c refers to the R channel histograms merging part B and part C of (2). Finally, the R channel histograms combining all parts in (2) are demonstrated in Fig. 5e. As can be seen in the first row of Fig. 5, for an image having mostly small pixel values, the histogram of its transformed result tends to lean to the right hand side while having very few small values. On the other hand, if an image contains lots of big pixel values, the histogram of its transformed result is evidently pushed towards the left hand side, as shown in the

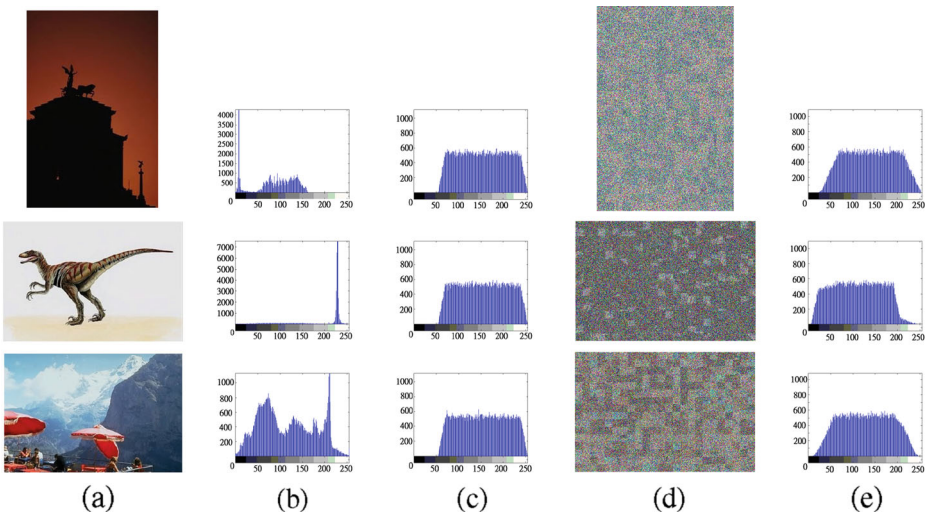


Fig. 5 **a** Three original images. **b** The R channel histograms of the original images shown in **(a)**. **c** The R channel histogram of the merged part B and part C of the proposed image transformation formula, (2). **d** The transformed results of **(a)**. **e** The R channel histograms of **(d)**

second row of Fig. 5. When an image has a relatively flat histogram, the histogram of its transformed result is then evenly distributed within the middle range, as shown in the third row of Fig. 5.

Note that the absolute value of W should be controlled not to be too small before such a scheme works. As the absolute value of W acts as the strength that an original pixel value pushes the transformed histogram towards a certain direction, a rather small absolute value of W means a weak pushing power, which makes little effect on the resulting histogram. For example, if W equals to 0.01, then for a pixel value as big as 200, the resulting histogram is still only slightly shifted, which can even be regarded as no effect if the bin size of the histogram is set to be larger than 2. Therefore, in our current implementation, we always ensure that the absolute value of W is bigger than 0.1.

4.2 Encryption domain image convolution

Some of the basic image processing techniques, such as blurring, smoothing, edge detection, and so on, can be achieved by *image convolutions*, which typically involve the conjunctions of several additive and multiplicative operations on the input images. Unlike other block-based image encryption methods, which introduce non-linear mappings to the pixel values, the proposed approach purely adopts *linear combinations* to image pixels during the encryption. As a consequence, it is straightforward to apply additive and multiplicative operations to the encrypted images under the proposed encryption framework. Assume that a client has already uploaded some private (encrypted) images to a remote server, and wishes to apply some convolutional image processing to the uploaded images. It will not be necessary for the client to derive the original image first, and then apply the desired convolutional image processing all on its own, as would be the cases in most traditional encryption/decryption schemes. Instead, in the proposed image encryption scheme, the remote server can execute the image convolution *directly on the encrypted images*, and then send back the convolved

results to the client. All the client needs to do is to perform the decryption as defined in (3). To ease the explanation, we use an example showing how multiplying two encrypted pixels with two different numbers and then adding them up together can be done in the encryption domain. It is straightforward to extend from two pixels to multiple pixels. First, let us re-write (2) and let the new equation represent the encryption of only one pixel. For a pixel, say A, its encrypted result is presented as

$$\begin{aligned} E_A = \quad & (W1) \cdot P_A + (W2 \cdot 255) \\ & + (W3 \cdot N_{A1}) + (W4 \cdot N_{A2}) \end{aligned} \tag{4}$$

Note that in the original equation the weightings, i.e. w_1 to w_6, stay the same for every pixel and every image, therefore we further combine their linear combinations into W_1 to W_4. For another pixel B, the equation can be derived in the same fashion:

$$\begin{aligned} E_B = \quad & (W1) \cdot P_B + (W2 \cdot 255) \\ & + (W3 \cdot N_{B1}) + (W4 \cdot N_{B2}) \end{aligned} \tag{5}$$

Then, multiplying pixels A and B with two different numbers, say C_1 and C_2, and adding the multiplied values together in the encryption domain can be written as the following:

$$\begin{aligned} C_1 \cdot E_A + C_2 \cdot E_B = \ & (C_1 \cdot W1) \cdot P_A + (C_2 \cdot W1) \cdot P_B + (C_1 + C_2) \cdot (W2 \cdot 255) \\ & + C_1 \cdot (W3 \cdot N_{A1}) + C_2 \cdot (W3 \cdot N_{B1}) \\ & + C_1 \cdot (W4 \cdot N_{A2}) + C_2 \cdot (W4 \cdot N_{B2}) \end{aligned} \tag{6}$$

Next, we treat (6) similarly as we did to (2) for deriving (3), that is, put what we want to get on the left hand side, and the others on the right hand side:

$$\begin{aligned} C_1 \cdot P_A + C_2 \cdot P_B = \ & \frac{C_1 \cdot E_A + C_2 \cdot E_B - (C_1 + C_2) \cdot (W2 \cdot 255)}{W1} \\ & - \frac{C_1 \cdot (W3 \cdot N_{A1}) + C_2 \cdot (W3 \cdot N_{B1})}{W1} \\ & - \frac{C_1 \cdot (W4 \cdot N_{A2}) + C_2 \cdot (W4 \cdot N_{B2})}{W1} \end{aligned} \tag{7}$$

Finally, through (7), it is obvious that in order to yield the value of $C_1 \cdot P_A + C_2 \cdot P_B$, the client only needs the server side to calculate and send back the value of $C_1 \cdot E_A + C_2 \cdot E_B$, while other computations can be done by the client side itself. As a result, the image convolution can be carried out in the encryption domain under the proposed image encryption scheme.

There are two small modifications to be made before such a scheme works. One is that the partitioning procedure during the encryption process makes the boundaries of each block disconnected with their original neighbors. This can be amended by storing some extra overlapped pixels along the boundaries of the blocks, where the width of the overlapped areas depends on the kernel size of the image convolution operator. For example, a 3×3 operator requires the overlapping areas to be at least one pixel wide. The other issue is that the white noise images N_1 and N_2 should be convolved by the same operator before the decryption. But we would like to remark that we can first calculate the convolved result of the initially given constant white noise image, N, and then synthesize the convolved N_1 and N_2. By doing this, for every decryption process, the convolution only needs to be executed once instead of twice. Even better, in the case that the operator of an image convolution is pre-defined, or when there are tons of images need to be applied by the same operator, pre-computing the convolved N not only will accelerate the entire process but also can reduce the client's computational load.

5 Experimental results & evaluation

The experiments are performed on a machine with an Intel Core 2 Quad CPU Q9505 2.83GHz and 3GB memory. The operations of the proposed image encryption, decryption, retrieval and convolution are implemented with MATLAB R2011b. There are 2 image datasets used in our experiments, called $D1$ and $D2$. D1 is the well-known *Wang Dataset* [37] which contains 1000 color images, and the image sizes are either 256×384 or 384×256. In the Wang Dataset, the images are classified into 10 classes with 100 images each, which make it a frequently used dataset for testing the performance of an content-based image retrieval system. $D2$ is composed of 65 color images collected by ourselves, and the images are scaled to the sizes of 640×480 or 480×640. In our experiments, $D1$ is mainly used to test the efficiency of the proposed approach on a relatively larger dataset, while $D2$ is used for confirming that the proposed method still offers decent performance even when facing with bigger image sizes.

5.1 Performance analyses

In the performance analyses, we analyzed the storage overhead introduced by the overlapped pixels in order to apply convolutions on the encrypted images. In addition, the time spent executing the operations on the client side is measured. Furthermore, the performance of encryption domain content-based image retrieval and image convolution under the proposed method is evaluated.

5.1.1 Storage overhead analysis

As mentioned, in order to provide the functionality of image convolution in the encryption domain, some extra overlapping pixels need to be stored in the transformed images. The width of the overlapped pixels will determine the overhead of the totally stored pixels in an image sub-block, which also depends on the size of the system-supported image convolution operation. The storage overhead ratio of an encrypted image is calculated according to the block size used in the permutation process. A larger permutation block size leads to a lower storage overhead ratio and corresponds to less processing time as well; however, it may also make the features in the original images more visually perceptible in their encrypted forms. This trade-off definitely needs to be balanced while setting up the configurations. Table 1 shows the results of encrypted image sizes and the storage overhead ratios under different image encryption setups. Due to the rapid evolution of cloud storage in both storage and network capacity, storage cost is a lower priority concern than privacy in most cases. As a result, the cost of the storage capacities is not expensive anymore, and acquiring large amount of data from a remote server should no longer be a serious problem.

5.1.2 Time analysis

The analyses discussed in this subsection are mainly based on the case where the permutation block size is 16×16. Table 2 presents the averaged encryption/decryption times for the images in $D1$ and $D2$. The numbers represent the averaged processing time of the proposed image encryption/decryption algorithm if we perform the image encryption/decryption process separately on every single image. Nevertheless, sometimes it is also possible to execute the encryption/decryption process in a batch mode. That is, we pre-load the initially created constant white noise image, N, that we use to generate the single-use N_1 and N_2. As the

Table 1 The variation of the image sizes before/after the proposed image encryption process with different configurations as well as the storage overhead ratios

Experimental Images in $D1$ (Image size: 384×256 or 256×384)			
	Width of the overlapped pixels		
	1	2	3
Permutation Block Size: 8×8	480×320 or 320×480	576×384 or 384×576	672×448 or 448×672
Storage Overhead Ratio	56 %	125 %	206 %
Permutation Block Size: 16×16	432×288 or 288×432	480×320 or 320×480	528×352 or 352×528
Storage Overhead Ratio	27 %	56 %	89 %
Permutation Block Size: 32×32	408×272 or 272×408	432×288 or 288×432	456×304 or 304×456
Storage Overhead Ratio	13 %	27 %	41 %

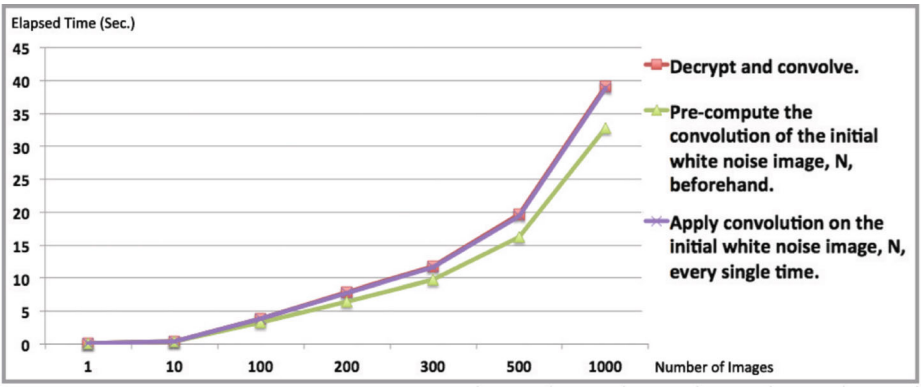
number shown in the parentheses in Table 2, this batch mode approximately doubles the efficiency on both the encryption and decryption processes.

The analysis also considered the time spent computing the image convolution. As described in the *Encryption Domain Image Convolution* sub-section, the convolved N_1 and N_2 could be derived by pre-computing the convolved version of the initial white noise image, N, either when the image convolution operation is pre-defined or when there are large number of images need to be processed by the same operation. Figure 6 demonstrates the comparison between the execution time with and without pre-convolving N. Furthermore, we include the time for directly decrypting the encrypted images and applying image convolutions in their original domain, i.e. the traditional solution. The experimental setups are as follows: 1) The images used are from $D1$. 2) The width of overlapped pixels is set

Table 2 The averaged encryption/decryption time (measured in seconds) of the images in $D1$ and $D2$ with the permutation block size set to 16×16

Width of Overlapped	Testing images in $D1$ (384×256 or 256×384)		Testing images in $D2$ (640×480 or 480×640)	
Pixels	Encryption	Decryption	Encryption	Decryption
P0	0.09(0.05)	0.06(0.03)	0.37(0.19)	0.26(0.07)
1	0.11(0.07)	0.07(0.03)	0.47(0.24)	0.31(0.08)
2	0.12(0.07)	0.08(0.03)	0.56(0.28)	0.36(0.09)
3	0.14(0.08)	0.09(0.04)	0.66(0.32)	0.42(0.10)

The numbers in the parentheses indicate the cases when the initially created white noise image, N, is pre-loaded to the memory buffer, which accelerates the encryption/decryption time significantly



Number of Images	1	10	100	200	300	500	1000
Decrypt and convolve.	0.04	0.38	3.91	7.85	11.78	19.7	39.05
Pre-compute the convolution of the initial white noise image, N, beforehand.	0.05	0.34	3.23	6.47	9.75	16.28	32.82
Apply convolution on the initial white noise image, N, every single time.	0.04	0.38	3.82	7.73	11.57	19.4	38.8

Fig. 6 A comparison of elapsed time among three approaches to achieve the goal of convolutional image processing in the encryption domain: 1) Decrypt the encrypted images to their plain domain, and then apply the image convolution. 2) Pre-compute the convolved initial white noise image, N, which is used for generating non-repeated N_1 and N_2 in advance. 3) Compute the convolved initial white noise image, N, every single time. As presented by the *green line*, if the convolved N can be pre-computed under the proposed image encryption scheme, not only the computational load of the client side is lowered, but also the processing time is accelerated

to 3, and the size of the applying kernel equals to 7×7. As can be seen in Fig. 6, when the client side needs to convolve N on its own in every single run, the average computation time would be very close to that of the traditional scheme. The only difference is that in the proposed method, the client has to obtain the convolved initial white noise image first, and then proceeds with the decryption process. As mentioned earlier, we can pre-calculate the convolved N beforehand if the convolution kernel has been pre-defined or when there are lots of images need to be processed at one time. This pre-computation allows the client side to only have to recover the random selected positions for synthesizing the non-repeated N_1 and N_2, before performing the inverse transformation to obtain the final results, thus reducing the overhead of the convolution processes. As shown by the green line in Fig. 6, the performance benefit of directly convolving the encrypted images increases with the number of consecutive images to be processed.

5.1.3 Encryption domain content-based image retrieval

Deselaers et al. [7] surveyed current state-of-the-art content-based image retrieval mechanisms, in which color features directly calculated using image pixel values [15, 16, 33, 34] are the most basic and effective features used in the research of image retrieval, such as color histograms and color moment. Other features like image textures [25, 29], shapes [2] and SIFT features [21] are also frequently adopted in various types of image retrieval applications. Instead of using precision and recall values, [7] adopted a measurement of so-called *"mean average precision (MAP)"*, which has the advantage of taking both precision and

recall values into account using only one number. To derive the *MAP* value, the average precision (AP) for a single query, q, is calculated first:

$$AP(q) = \frac{1}{N_R} \sum_{n=1}^{N_R} P_q \cdot R_n, \tag{8}$$

where $P_q \cdot (R_n)$ is the precision rate after the n-th relevant image was recalled and N_R is the total number of relevant images to the query. Then, the MAP value is computed as the mean of the average precision scores over all queries:

$$MAP = \frac{1}{|Q|} \sum_{q \in Q} AP(q), \tag{9}$$

where Q is the set of input queries. They also applied a classification error rate (*ER*) to specify how well a dissimilarity measure can find the most relevant image correctly. The *ER* value is computed as:

$$ER = \frac{1}{|Q|} \sum_{q \in Q} \begin{cases} 0, & \text{if the most similar image is relevant.} \\ 1, & \text{otherwise.} \end{cases} \tag{10}$$

As recommended in [7], the simple color histogram, although often used as a baseline, performs surprisingly well for the purpose of image retrieval. In many cases, it even outperforms most of the other image features. We then adopted the same color-histogram-based image retrieval method that used by [7] on the encrypted images and compared its retrieval performance with the ones conducted by [7] using the Wang Dataset [37]. The distance metric used in our experiment was Jensen-Shannon divergence (JSD), which was also suggested in [7]. The comparison is presented in Table 3, where the first row indicates the *ER* and *MAP* values yielded by the proposed approach, while the following 19 rows show the corresponding *ER* and *MAP* values of the methods listed in [7]. Although the proposed encryption domain content-based image retrieval obtained a 10 %-loss in terms of *MAP* as compared with the approach using RGB color histogram in the plain domain, we argue that competitive performance can still be produced as compared with other approaches. Among the 19 approaches addressed in [7], except for *"RGB color histogram"*, only *"LF patches histogram"* and *"Inv. feature histogram (monomial)"* yield better results than the proposed method in both *MAP* and *ER* values. The proposed approach even entirely dominates more than half of the other methods, i.e. yielding a higher *MAP* value and a lower *ER* value, not to mention the proposed method also provides additional image content protection.

We also measured the precision rate when top K similar images are retrieved as in traditional image retrieval performance evaluation:

$$Precision(q) = \frac{NR}{NT}, \tag{11}$$

where NR refers to the number of relevant image being retrieved, while NT means to the total number of image to be retrieved, i.e. the value of K. Then, the averaged precision rate is derived by computing the mean of precision rates over all queries. In this experiment, every image in the Wang Dataset [37] is used as a query to obtain the final averaged precision rate. Table 4 provides the averaged precision rate (%) of the datasets for the top K similar retreived images with and without the proposed image encryption. It shows that there is a 4 % $\sim$ 10 % difference on the averaged precision rates between the encryption domain and the plain domain image retrieval results. Again, this experiment was comparing the proposed method with the approach using RGB color histogram in the plain domain, which

Table 3 Comparison of the image retrieval performance between the proposed method and the ones listed in [7]

Feature	Error rate (ER)(%)	Mean average precision (MAP)(%)
The proposed encryption domain RGB color histogram	21.0	40.5
RGB color histogram	16.9	50.5
LF SIFT global search	37.2	38.3
LF patches histogram	17.9	48.3
LF SIFT histogram	25.6	48.2
Inv. feature histogram (monomial)	19.2	47.6
MPEG7: scalable color	25.1	46.7
LF patches signature	24.3	40.4
Gabor histogram	30.5	41.3
32*32 image	47.2	37.6
MPEG7: color layout	35.4	41.8
Xx32 image	55.9	24.3
Tamura texture histogram	28.4	38.2
LF SIFT signature	35.1	36.7
Gray value histogram	45.3	31.7
LF patches global	42.9	30.5
MPEG7: edge histogram	32.8	40.8
Inv. feature histogram (relational)	38.3	34.9
Gabor vector	65.5	23.7
Global texture feature	51.4	26.3

The dataset used is the Wang Dataset [37]. The first row displays the *ER* and *MAP* values yielded by extracting RGB color histogram from the encrypted images. The following 19 rows indicate the corresponding *ER* and *MAP* values of the methods listed in [7]

could be considered the most effective method with the Wang Dataset [37] according to [7]. If other techniques/features were adopted, the difference might become less noticeable.

The above experiments demonstrate that the image retrieval performance of the proposed approach matches with general content-based image retrieval algorithms. On top of that, the

Table 4 The averaged precision rate (%) between the encrypted images and the plain images when the top K similar images are retrieved

Top K results	Plain domain (%)	Encryption domain (%)
K=3	86.4	82.2
K=5	82.1	73.9
K=10	75.5	64.3

proposed method also offers additional image content protection as well as the functionality of encryption domain image convolution. In exchange with these extra advantages the proposed method provides, we believe a bit of accuracy drop should be a quite acceptable trade-off for users with security concerns.

5.1.4 Encryption domain image convolution

As mentioned in the previous section, some of the basic image processing techniques, such as blurring, smoothing, edge detection, and so on, can be achieved by image convolutions. In the proposed image transformation scheme, such operations can be applied directly on the transformed images. Although some rounding errors may be introduced, we show that the induced errors are too small to be noticed. Figure 7 demonstrates examples of performing three basic image processing operations, i.e. blurring, sharpening, and edge detection, under the proposed image encryption scheme. In addition, we also compared the pixel value differences of the image convolution results between applying the convolutions in the plain domain and in the proposed encryption domain. As can be seen in Fig. 7f, although there are some small errors due to the truncation of floating points during the encryption/decryption process, the differences can almost be neglected and are hardly visible.

5.2 Security analysis

This section discusses the degree of security that the proposed image encryption approach can reach. For an image encryption system to be considered as robust, its encrypted outcomes should be regarded both statistically random and computationally intractable. The following analyzes will then be mainly focused on these two aspects.

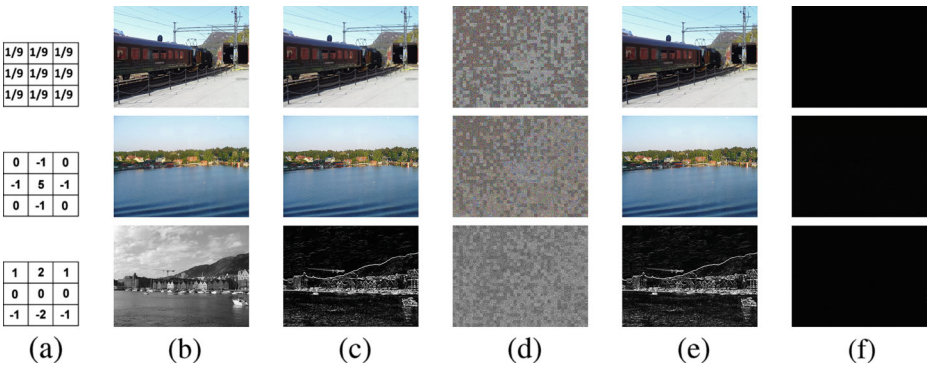


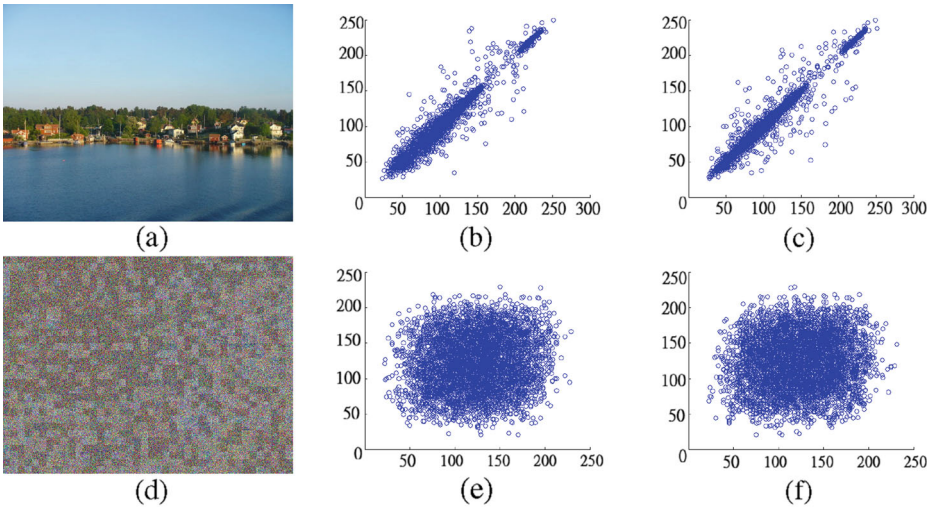
Fig. 7 Comparing the results of image convolutions in the plain domain and in the proposed encryption domain. Three basic image processing techniques, blurring, sharpening, and edge detection, are involved and the results are arranged in rows. **a** The convolution kernels used. **b** Input images. **c** Image convolution in the plain domain. **d** The encrypted images. **e** Image convolution applied on **(d)**, and followed by the decryption. **f** The pixel-wise difference between **(c)** and **(e)**. Note that, in the edge detection example (*third row*) we only applied the horizontal Sobel operator. Moreover, we did not set a threshold to identify the edge pixels. Therefore, the convolved results, **(c) and (e)**, show the absolute intensity after the convolution rather than the detected edges while intensity values over 255 are set to 255

5.2.1 Statistically

There are two well-known coefficients, *correlation* and *information entropy*, that are often used for verifying the randomness of a given distribution. We adopted the correlation coefficient to measure the correlation between two adjacent image pixels:

$$\frac{n\sum(x_i y_i) - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - \left(\sum x_i\right)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2}} \tag{12}$$

where n refers to the number of pairs of adjacent pixels selected, and x_i and y_i represent the intensity values of the two chosen neighboring pixels. A higher correlation value means that the selected pairs of pixels are highly correlated with each other, and such a phenomenon usually happens on a normal, non-encrypted image. On the other hand, a well-encrypted image typically yields a low correlation value. In our experiment, we set the permutation block size as 16×16 and randomly select 5000 pairs of adjacent pixels vertically and horizontally for the computation of the correlation coefficients. Figure 8 illustrates the pixel value distributions and the correlation coefficients of an example image before/after its



	Plain Image	Encrypted Image	D1	D2
Vertical Correlation	0.9902	0.088	0.083	0.093
Horizontal Correlation	0.9912	0.084	0.082	0.096

(g)

Fig. 8 An example of the correlation coefficients computed from an original image and its associated encryption. The image is encrypted with permutation block size equals to 16×16 and no overlapped pixels on the sub-block boundaries. And the number of pairs of selected neighboring pixels is 5000. **a** An original image. **b & c** The distribution of the adjacent pixels randomly picked from the original image vertically and horizontally, respectively. **d** The encrypted image. **e & f** The distribution of the adjacent pixels randomly picked from the encrypted image vertically and horizontally, respectively. **g** The correlation values of the original image and the encrypted image Also, the averaged correlation values of all the encrypted results of the images in $D1$ and $D2$ are listed

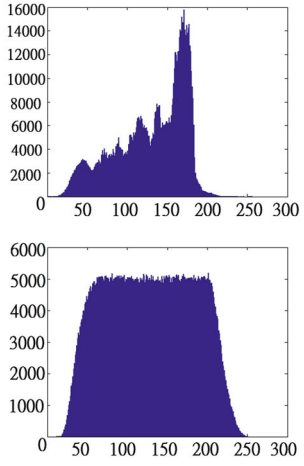
encryption. Moreover, the averaged correlation values of all the encrypted results of the images in $D1$ and $D2$ are also provided in Fig. 8g.

Information entropy is another good estimator to gauge the uncertainty level of a specific distribution. For an image, it is used to judge the degree of randomness regarding the distribution of the pixel values. The entropy value of an image is calculated by the following formula:

$$-\sum_{k=0}^{255} p(k) log_2 p(k) \tag{13}$$

where k refers to the RGB values of an input image (0...255), and $p(k)$ is the probability of the occurrence of RGB value k. Through the computation of the entropy value by (13), a truly random white noise image leads to an entropy value of 8. However, achieving such an ideal image encryption result poses a great challenge to most of the currently existing image encryption algorithms. Furthermore, as the proposed image encryption applies linear combinations to several images, it is less possible to map the original pixel values to the extremely high/low values, thus affecting the values of the entropy. Even so, the proposed method still produces comparable outcomes. As can be seen in Fig. 9a and b, the histogram of the original image contains several peak values. In contrast, the histogram of the encrypted image is mostly uniformly distributed in the middle range of the entire histogram, as shown in Fig. 9c and d. The overall statistics of the entropy values from all the encrypted results of the images in $D1$ and $D2$ are displayed in Fig. 9e.

Except for measuring the correlation and entropy values of all the images in $D1$ and $D2$, we also compared the correlation and entropy values which can be yielded by the proposed method with other block-based transformation image encryption approaches. Figure 10a and c show two images, *"Lenna"* and *"Barbara"*, well-known in the image processing



	Entropy
Plain Image	7.1748
Encrypted Image	7.6650
D1	7.7040
D2	7.7120

Fig. 9 An example of the information entropy calculated from an original image and its associated encryption. The image is encrypted with permutation block size equals to 16×16 and no overlapped pixels on the sub-block boundaries. **a** An original image. **b** The histogram of the pixel values in the original image. **c** The encrypted image. **d** The histogram of the pixel values in the encrypted image. **e** The entropy values of the original image and the encrypted image. Also, the averaged entropy values of all the encrypted results of the images in $D1$ and $D2$ are listed

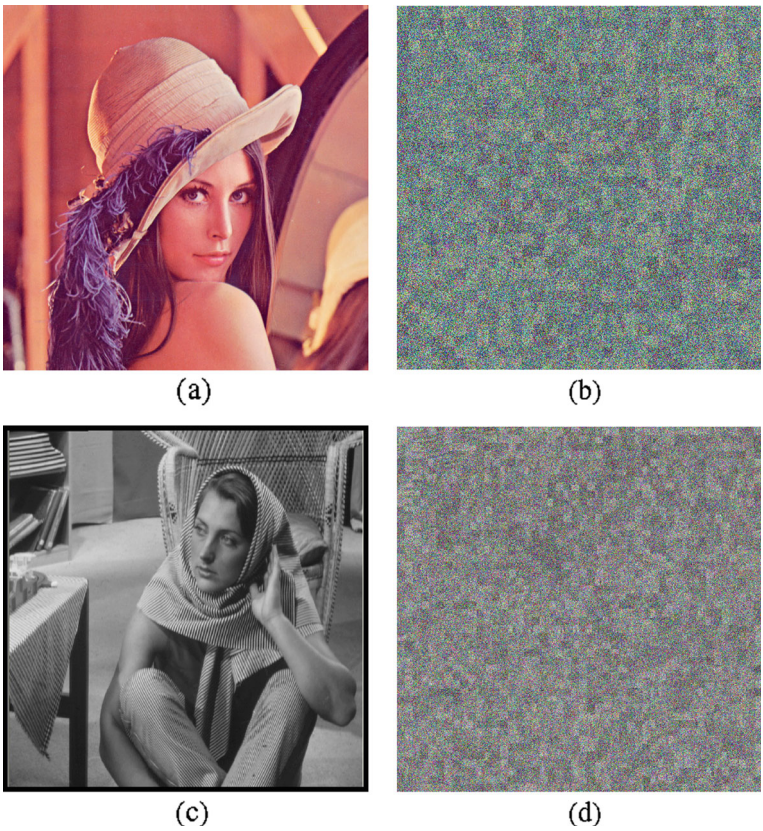


Fig. 10 **a & c** The two well-known images, *"Lenna"* and *"Barbara"*, in the image processing community that are used for comparing the entropy and correlation values. **b & d** The corresponding encrypted results of **(a) & (c)**

community, which are used for the comparison, while Fig. 10b and d are their corresponding encrypted results by the proposed method. The comparison results can be seen in Table 5. Although the proposed method does not produce encrypted images with as high entropy and low correlations values as other block-based approaches do, it is still worth mentioning again that the other techniques do not support any other functionalities, such as image retrieval and convolution, while only focusing on achieving stronger security to the encryption results.

To further evaluate the effectiveness of the proposed image encryption method, we also adopted the fast Fourier Transform (FFT) to see if the attackers could find obvious vulnerability through the frequency domain. We randomly selected five image from $D1$ and applied FFT to them as well as to their encrypted results. As can be seen in Fig. 11a, the five chosen images have rather different content and behave differently in their FFT spectrums, as shown in Fig. 11b. However, in Fig. 11c and d we can observe that in their corresponding encrypted results the FFT spectrums look very similar to each other, which means they are considered to be similar in the frequency domain. Therefore, it should not be a trivial work when an attacker tries to learn more information about the encrypted images through a frequency domain analysis.

Table 5 A comparison of entropy and correlation values derived by the proposed method and other approaches

Image	Entropy	Horizontal correlation	Vertical correlation
Lenna (Plain)	7.5875	0.9883	0.9958
Lenna encrypted by the proposed method	7.7484	0.0354	0.0533
Lenna encrypted by [12]	7.9313	0.0045	-0.0188
Lenna encrypted by [40]	n/a	0.0007	0.0016
Lenna encrypted by [31]	7.99	0.0001	0.0016
Barbara (Plain)	7.3475	0.9976	0.9924
Barbara encrypted by the proposed method	7.7062	0.0748	0.0552
Barbara encrypted by [17]	7.99	0.0096	0.0038
Barbara encrypted by [31]	7.99	0.0018	0.0013

5.2.2 Computationally

Although the provided security of the proposed method may not be as high as traditional encryption schemes, we still want to claim that it is still computationally expensive to some extent for one to decipher the images that are encrypted by the proposed approach. In order to do that, we discuss the involved complexity faced by an attacker, who does not possess the correct encryption keys, when he/she tries to uncover the original image content from the encrypted images. In general, two attack models are commonly adopted in cryptanalysis, namely *cipher-text only attack* and *known plain-text attack*.

Cipher-text only attack (COA) This attack model assumes that an attacker has obtained the access to some samples of the cipher-texts, which refer to the encrypted images. There are two main operations adopted by the proposed method that make the encrypted image secure. One is the random permutation of the image sub-blocks, which disperses the image features. The other is the involvement of white noise images that eliminates the correlations among adjacent pixel values. As described in the *Image Encryption Scheme* Section, resolving the permutation of the image sub-blocks would require exhaustive searches for all possibilities if finding the right matches among adjacent sub-blocks cannot be guaranteed until the entire image puzzle has been solved. For example, a 640×480 sized image with 32×32 pixels per sub-block will finally be partitioned into 300 sub-blocks, which means there are 300! ways to arrange the sequences of all the sub-blocks potentially. Not to mention that in our case the answer to the image puzzle is a vague and obscured image, so one may not be 100 % sure even when the correct permutation has already been derived. Furthermore, for one who wishes to remove the white noises from an encrypted image, trying every possible combination of parameters and random white-noised images is still required. As can be observed in (2), even if the weightings can all be ignored, the pixel values of the white noise images are still needed to obtain the correct original pixel values. According to section 9.4 in [9], the total workload for the attacker to break a 2^{20}-bytes random block would potentially end up being 2^{113} steps, which means that the complexity of enumerating all possible white noise images of a 32×32 image sub-block can only be more expensive.

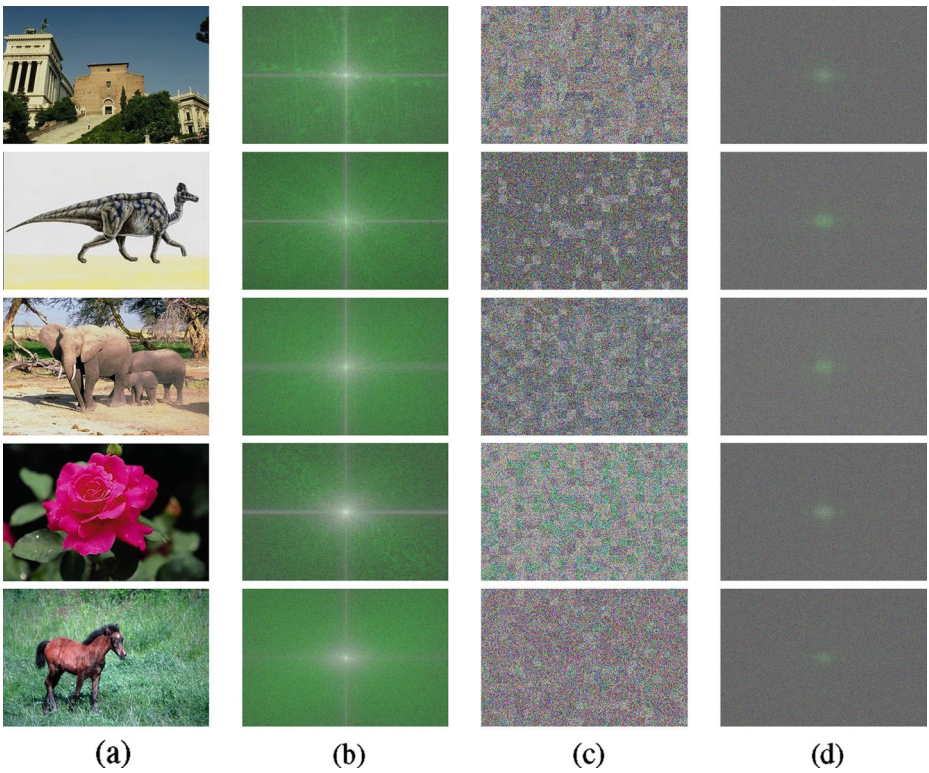


Fig. 11 **a** Five randomly chosen images from $D1$. **b** The FFT spectrums of **(a)**. **c** The corresponding encrypted results of **(a)**. **d** The FFT spectrums of **(c)**

Known plain-text attack (KPA) This is an attack model that presumes an attacker has obtained a set of both the plain-text and its associated cipher-text. That is, an attacker may have already obtained some of the encrypted images and have correctly identified what their corresponding original images are. In the proposed framework, to guess the right encryption key with the original image and its encrypted version is approximately the same as only knowing the encrypted image, while one has to first resolve the permutation of the image sub-blocks as it might take exhaustive search of all possible combinations. As the example mentioned in the *COA analysis*, for a puzzle of 640×480 sized image with 32×32 pixels per sub-block will eventually take up to 300! trials. Therefore, finding the mapping of a sub-block in the encrypted image with its original patch becomes intractable. In addition, the proposed approach adopts different permutation sequences and non-repeated white noise images to encrypt every single image. As a result, in the circumstance that only a small amount of plain-image and cipher-image sets are known to the attacker, the complexity of breaking the proposed image encryption scheme under this attacking model still remains the same as in the COA model. Even if the attacker has cracked all the permutation sequences of the obtained plain-image and cipher-image sets, which means some patches of the white noise images may possibly be compromised. However, using the proposed method a 640×480 sized white noise image with sub-block size being set to 32×32 could finally yield out a total number of $(608 \times 448)^2 \cong 2^{36}$ possible white noise image patches, while every encrypted image randomly acquires only 300 patches from them. In other words, it may

potentially take up to $2^{36} \div 300 \cong 2^{28}$ plain-image and cipher-image sets for an attacker to re-assemble the white noise image used by the client. Furthermore, we also like to remark that this number is under the assumption that all the permutation sequences have already been compromised.

5.3 Limitations

There exist some limitations in the proposed image encryption scheme. First, during the image transformation process, if one wishes to spend less storage capacity, he/she may want to apply some compression to the resulting encrypted images. The compression will cause some information loss, thus inducing some noises in the decrypted results. However, one should note that if the images are compressed with a more high quality compression approach, the quality of the inversely transformed images would be more acceptable. Figure 12 displays the comparisons of the decryption results while using different levels of JPEG image compression qualities or even with no compression. As can be seen in Fig. 12d, when the highest compression quality is applied, the yielded results are hardly noticed to be different from the results with no compression involved, as shown in Fig. 12b.

Another limitation is related to the convolution kernel applied. The proposed method yields results with high quality if a constant image convolution operator is applied to every image pixel. However, in some convolutional image processing, it involves pixel value

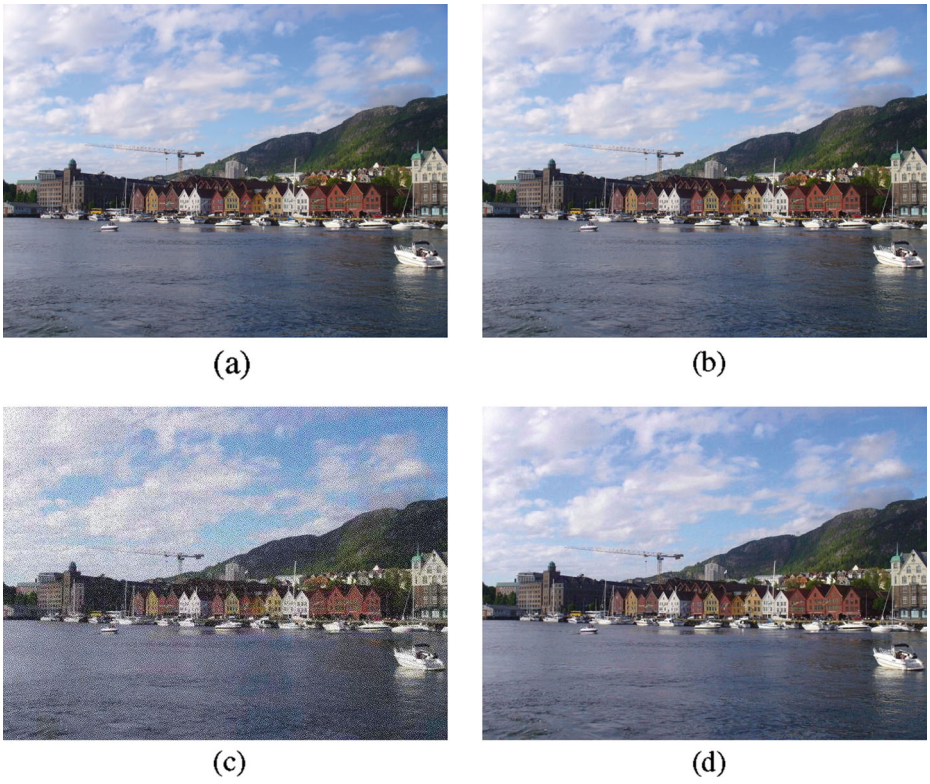


Fig. 12 **a** Original image. **b** The decryption result without any compression. **c** The decryption result with a relatively low JPEG compression quality. **d** The decryption result with the highest JPEG compression quality

comparisons to decide the final values of the convolution kernel for each pixel. For example, while applying the bilateral filtering or the Canny edge detector, differences in pixel values or magnitudes of pixel gradients need to be calculated first to decide the values in the convolution kernel for a pixel. In the proposed image encryption framework, comparing the original pixel values in the encryption domain still remains an unsolved issue. As a result, we currently can only handle the image convolution operators that apply the same convolution kernel to all image pixels.

Finally, although a one-time pad scheme is used to create the user-specified secret key, the random selection of initial white noise image patch still has shortened the actual key-size. As the number of encrypted images increases to a certain amount, the user might want to consider re-generating a new white noise image in order to keep the image security at a certain level.

6 Conclusion

We have proposed a image encryption algorithm under the concept of block-based transformations. The proposed image encryption scheme makes content-based image retrieval and convolutional image processing in the encryption domain feasible. According to these features of the encryption, a secure on-line image storage database system can be easily constructed for supporting computation delegation and image retrieval while at the same time preserving the privacy of the uploaded images. Experimental results show that the proposed approach is not only secure but also provides a good performance in terms of the involved functionalities. Although the computational complexity to crack the proposed method may not be as high as traditional cryptographic encryption algorithms, we still believe that adopting the proposed approach is a worthy trade-off and could fulfill the need for users as the additionally supplied functionalities of image retrieval and image convolution are made applicable in the encryption domain.

There are, however, still many potential directions that we can pursue in the future. One possible direction is to extend the encryption framework to videos. A straightforward generalization from an image to a video may not work directly as temporal coherence in a video may give out hints in its encrypted form. Hsu et al. [14] proposed a privacy-preserving secure SIFT method in the homomorphic encryption domain, and presented a secure comparison method that can be conducted in the encryption domain. However, due to the homomorphic encryption, some of detected SIFT features are slightly different from what they are in the original domain. We would like to study the way of comparing encrypted data values under the proposed encryption scheme. If we succeed, then many other applications can also be realized. For example: computing the saliency maps, performing median filtering on images or applying some of the non-linear image convolution operators, etc. Cancellaro et al. [3, 4], Bianchi et al. [1], Subramanyam et al. [35], and Qin and Zhang [30] proposed to embed watermark or secret information into the encrypted image while checking out the watermark or secret information can be done without deciphering, so that both the authenticity and privacy of image are maintained. This feature might be favorably desired under the proposed framework, because users may want to protect the ownership and to verify the integrity of the uploaded images in a large image storage database system.

Acknowledgments The authors thank the anonymous reviewers' spending valuable time reviewing and giving out useful comments. We also would like to thank Nick Leaf for proof-reading the paper and providing

suggestions on how to address the reviewers' comments. This work was supported in part by the Ministry of Science and Technology, Taiwan under the grant NSC 101-2221-E-011-150-MY3.

References

1. Bianchi T, Piva A, Barni M (2009) Encrypted domain dct based on homomorphic cryptosystems. EURASIP J Inf Secur:1:1–1:12. doi:10.1155/2009/716357
2. Bober M (2001) MPEG-7 Visual Shape Descriptor. IEEE Transactions on Circuits and Systems for Video Technology 11(6):716–719
3. Cancellaro M, Battisti F, Carli M, Boato G, Natale FGBD, Neri A (2008) A joint digital watermarking and encryption method. SPIE 6819:68191C
4. Cancellaro M, Battisti F, Carli M, Boato G, Natale FGBD, Neri A (2011) A commutative digital image watermarking and encryption method in the tree structured haar transform domain. Signal Process Image Commun 26(1):1–12
5. Chu KY, Kuo YH, Hsu WH (2013) Real-time privacy-preserving moving object detection in the cloud. In: Proceedings of the 21st ACM international conference on multimedia, MM '13, pp 597–600
6. Demaine ED, Demaine ML (2007) Jigsaw puzzles, edge matching, and polyomino packing: connections and complexity. Graphs and Combinatorics 23:195–208
7. Deselaers T, Keysers D, Ney H (2008) Features for image retrieval: an experimental comparison. J Image Retr 11(2):77–107
8. Erkin Z, Piva A, Katzenbeisser S, Lagendijk RL, Shokrollahi J, Neven G, Barni M (2007) Protection and retrieval of encrypted multimedia content: When cryptography meets signal processing. EURASIP Journal on Information Security 7(2):1–20
9. Ferguson N, Schneier B, Kohno T (2010) Cryptography engineering: design principles and practical applications. Wiley
10. Fisher RA, Yates F (1938) Statistical tables for biological, agricultural and medical research, 3rd edn. Oliver & Boyd, London
11. Fontaine C, Galand F (2007) A survey of homomorphic encryption for nonspecialists. EURASIP J Inf Secur 2007:1–15
12. Gautam A, Panwar M, Gupta P (2011) A new image encryption approach using block based transformation algorithm. Int J Adv Eng Sci Tech 8(1):90–96
13. Gentry C (2010) Computing arbitrary functions of encrypted data. Commun ACM 53(3):97–105
14. Hsu CY, Lu CS, Pei SC (2011) Homomorphic encryption-based secure SIFT for privacy-preserving feature extraction. SPIE 7880:788005
15. Huang ZC, Chan PK, Ng WY, Yeung DS (2010) Content-based image retrieval using color moment and gabor texture feature. In: Proceedings of the ninth international conference on machine learning and cybernetics, pp 719–724
16. Jeong S, Won CS, Gary RM (2004) Image retrieval using color histograms generated by gauss mixture vector quantization. Comp Vision Image Underst 94(1–3):44–66
17. Jolfaei A, Mirghadri A (2010) An image encryption approach using chaos and stream cipher. J Theor Appl Inf Technol 19(2):117–125
18. Kelsey J, Schneier B, Ferguson N (2000) Yarrow-160: notes on the design and analysis of the yarrow cryptographic pseudorandom number generator. In: Proceedings of the 6th annual international workshop on selected areas in cryptography, SAC '99, pp 13–33
19. Lathey A, Atrey PK (2015) Image enhancement in encrypted domain over cloud. ACM Trans Multimedia Comput Commun Appl 11(3):38:1–38:24. doi:10.1145/2656205
20. Lathey A, Atrey PK, Joshi N (2013) Homomorphic low pass filtering on encrypted multimedia over cloud. In: ICSC. IEEE, pp 310–313
21. Lowe DG (1999) Object recognition from local scale-invariant features. In: International conference on computer vision, pp 1150–1157
22. Lu W, Swaminathan A, Varna AL, Wu M (2009) Enabling search over encrypted multimedia databases. In: SPIE/IS&T media forensics and security, vol 7254. SPIE, p 725418
23. Lu W, Varna AL, Swaminathan A, Wu M (2009) Secure image retrieval through feature protection. In: IEEE conference on acoustics, speech and signal processing, pp 1533–1536
24. Maniccam S, Bourbakis N (2001) Lossless Image Compression and Encryption using SCAN. Pattern Recognit 34:1229–1245
25. Manjunath B, Ma W (1996) Texture features for browsing and retrieval of image data. IEEE Transactions on Pattern Analysis and Machine Intelligence 18(8):837–842

26. Melchor CA, Fau S, Fontaine C, Gogniat G, Sirdey R (2013) Recent advances in homomorphic encryption: a possible future for signal processing in the encrypted domain. IEEE Signal Process Mag 30(2):108–117
27. Mitra A, Rao YVS, Prasanna SRM (2006) A new image encryption approach using combinational permutation techniques. J Comput Sci 1(1):127–131
28. Naehrig M, Lauter K, Vaikuntanathan V (2011) Can homomorphic encryption be practical? In: Proceedings of the 3rd ACM workshop on cloud computing security workshop, CCSW '11, pp 113–124
29. Park M, Jin JS, Wilson LS (2002) Fast content-based image retrieval using quasi-gabor filter and reduction of image feature dimension. In: IEEE southwest symposium on image analysis and interpretation, pp 178–182
30. Qin C, Zhang X (2015) Effective reversible data hiding in encrypted image with privacy protection for image content. J Vis Commun Image Represent 31:154–164. doi:10.1016/j.jvcir.2015.06.009
31. Rakesh S, Ajitkumar AK, Shadakshari BC, B A (2012) Image encryption using block based uniform scrambling and chaotic logistic mapping. Int J Crypt Inf Secur (IJCIS) 2(1):49–57
32. Shamir A. (1979) How to share a secret. Commun ACM 22(11):612–613
33. Sharma N, Rawat P, Singh J (2011) Efficient CBIR using color histogram processing. Signal & Image Process Int J (SIPIJ) 2(1):94–112
34. Shih J, Chen LH (2002) Colour image retrieval based on primitives of colour moments. In: Proceedings of the 5th international conference on recent advances in visual information systems, pp 88–94
35. Subramanyam A, Emmanuel S, Kankanhalli M (2012) Robust watermarking of compressed and encrypted jpeg2000 images. IEEE Trans Multimed 14(3):703–716. doi:10.1109/TMM.2011.2181342
36. Upmanyu M, Namboodiri AM, Srinathan K, Jawahar CV Efficient privacy preserving video surveillance. In: ICCV. IEEE, pp 1639–1646
37. Wang JZ, Li J, Wiederhold G (2001) Simplicity: semantics-sensitive integrated matching for picture libraries. IEEE Trans Pattern Anal Mach Intell 23:947–963
38. Wang LT, McCluskey EJ (1988) Linear feedback shift register design using cyclic codes. IEEE Trans Comput 37(10):1302–1306
39. Yao AC (1982) Protocols for secure computations. In: Proceedings of the 23rd annual symposium on foundations of computer science, SFCS '82, pp 160–164
40. Yoon JW, Kim H (2010) An image encryption scheme with a pseudorandom permutation based on chaotic map. Commun Nonlinear Sci Numer Simul 15(12):3998–4006
41. Younes MAB, Jantan A (2008) An image encryption approach using a combination of permutation technique followed by encryption. Int J Comput Sci Netw Secur 8(4):191–197
42. Zeng W, Lei S (2002) Efficient frequency domain selective scrambling of digital video. IEEE Trans Multimed 5:118–129
43. Zhang X, Cheng H (2014) Histogram-based retrieval for encrypted jpeg images. In: 2014 IEEE china summit international conference on signal and information processing (ChinaSIP), pp 446–449. doi:10.1109/ChinaSIP.2014.6889282

Jia-Kai Chou received his Ph.D. degree in information management at National Taiwan University of Science and Technology in 2013. His research interests involve computer graphics, multimedia systems, and information visualization.

Chuan-Kai Yang received his Ph.D. degree in computer science from Stony Brook University, USA, in 2002, and his M.S. and B.S. degree in computer science and in mathematics from National Taiwan University in 1993 and 1991, respectively. He is currently a professor of the information management department at National Taiwan University of Science and Technology. His research interests include computer graphics, scientific visualization, multimedia systems, and computational geometry.

Hsing-Ching Chang received her Ph.D. degree in information management from National Taiwan University of Science and Technology, Taiwan, in 2008, her M.S. degrees in computer science and physics from Old Dominion University, Virginia, USA, in 1991 and 1989, respectively, and her B.S. degree in physics from Tamkang University, Taiwan, in 1984. She has been an associate professor in Chungyu Institute of Technology, Taiwan, since 2009. Her research interests include computer graphics, multimedia systems, computer animation and algorithms.