

Article

Hamming Code Strategy for Medical Image Sharing

Li Li ^{1,*}, Ching-Chun Chang ², Junlan Bai ³, Hai-Duong Le ⁴, Chi-Cheng Chen ^{5,*} and Teen-Hang Meen ⁶ 

¹ Institute of Computer Graphics and Image Processing, Hangzhou Dianzi University, Hangzhou 310018, China

² Department of Computer Science, University of Warwick, Coventry CV47AL, UK; ching-chun.chang@warwick.ac.uk

³ School of Electronic Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China; whitejunlan@foxmail.com

⁴ School of Computer Science and Engineering, International University, Vietnam National University HCMC, Ho Chi Minh 800010, Vietnam; duonghaile@gmail.com

⁵ Information and Engineering College, Jimei University, Fujian 361021, China

⁶ Department of Electronic Engineering, National Formosa University, Yunlin 632, Taiwan; thmeen@nfu.edu.tw

* Correspondence: lili2008@hdu.edu.cn (L.L.); ncue.anko@gmail.com (C.-C.C.)

Received: 8 December 2019; Accepted: 13 January 2020; Published: 19 January 2020



Abstract: In medical practice, the scanned image of the patient between the patient and the doctor is confidential. If info is stored on a single server and the server is successfully attacked, it is possible to expose confidential information. Password encryption and data authentication are commonly used to protect patient data, however, encryption and data authentication are computationally expensive and take time to execute on a mobile device. In addition, it is not easy for the patient details related to medical images to leak if the hacked image are not visual. Therefore, in this paper, we propose a way to make medical images remain untouched in this sense. We use our method to quickly create two shadows from two medical images and store them on two servers. Revealing a shadow image does nothing to compromise the confidentiality of a patient's health. This method is based on Hamming code. With low computational cost, the proposed scheme is suitable for tablet, pamphlets and other mobile devices.

Keywords: medical image; image sharing; hamming code

1. Introduction

Medical images play a major role in precise and detailed diagnoses nowadays. Doctors rely on them to figure out patients' illnesses and devise treatment programs. The images contain a lot of information regarding the patients' health conditions.

In most countries, the law mandates the confidentiality of patients' medical records. Medical images are, thus, only made available to the doctors involving in the treatment of a patient. Image confidentiality is often achieved by employing cryptographic encryption and data authentication mechanisms [1–4]. However, these methods are known to require a large amount of computing power, and encryption/decryption takes a long time to execute, especially on power-limited mobile devices such as tablets and phablets. In return, a swift response cannot be provided to doctors.

It is common and convenient for doctors to use mobile devices to access patients' medical information [5]. Therefore, there is a pressing demand for a faster secure mechanism that can ensure confidentiality of medical images and is suitable for mobile devices.

In the recent decades, some researches related to protecting privacy information for mobile services on different techniques, such as the redesign of the architecture of network [6], the non-interactive privacy-preserving protocol for image similarity computation [7], the data sharing protocol by using a new cryptographic primitive named online/offline attribute-based proxy re-encryption, and the transform key technique [8], have been developed. Also, some encryption methods based on cryptography, such as homomorphic encryption [9], elliptic curve cryptography based encryption [10] and chaotic oscillation theory-based encryption [11,12], were designed and applied to provide the confidentiality of patients' health information.

On the other hand, one characteristic of medical images is their high level of accuracy and detail. If a medical image is made blurred and loses its details, it will be useless. Therefore, instead of employing costly encryptions, we study how to make stored medical images less meaningful to the hackers who, somehow, successfully break into a database full of medical images.

In this paper, we propose a scheme, which creates shadow images (referred to as shadows) from medical images and stores them in different databases. The original images are only retrievable if all the shadows are collected. Individually, a shadow would not visually reveal any information regarding a patient's condition. This would satisfy the requirement for medical images. The proposed scheme is based on Hamming code, which has been used extensively in image processing [13–15]. Its encoding and decoding processes are very efficient, thus, it can help our scheme achieve low computational costs and guarantee a fast response when deployed on mobile devices.

2. Related Works

Image sharing has been studied extensively over the last decade. Most are rooted in the secret sharing concept. The very first secret sharing schemes were proposed by both Shamir and Blakely back in 1979 [16,17]. These schemes solve the problem of sharing a secret key among many participants. The trivial solution for secret key sharing is to provide each participant with a copy of the said key. However, the security of the secret key is not assured if only one copy is compromised. The solution is to split the secret into smaller pieces and give each participant a piece. The original secret will be fully reconstructed when all the required pieces are summoned.

Shamir's scheme [17] is a (k, n) threshold secret sharing scheme that divides the secret into pieces and requires at least k pieces out of n to reconstruct the secret. Because the scheme is based on Lagrange's interpolation modulo as the prime number, its direct application to greyscale images would introduce distortion to the reconstructed images as a result of truncating the pixels whose values are greater than the aforementioned prime number (in practice, they are truncated by 250). In 2002, Thien and Lin [18] proposed a solution for that problem allowing image sharing without loss. Their method is to preprocess the original image to normalize all the pixels to the range of 0 to 250 so that they can use interpolation modulo 251 to produce shadows. In addition, the shadows constructed this way are smaller in size compared to their original images. In 2006, Wang and Su [19] employed both the interpolation and Huffman coding to produce even smaller shadows. Later, Wang and Shyu [20] proposed a scheme that can be used to reduce the size of shadow images by half. The advantage of the smaller size of shadow images is obvious—the smaller the size, the less storage is required, especially when the shadows are kept on portable storage devices like USB drives. For colour images, Change et al. [21] developed a colour image sharing scheme based on the gradual search algorithm [22] and Shamir's secret sharing [17]. Randomization was introduced into shadows to raise the security level. In 2009, Tsai et al. [23] combined neural network and visual secret sharing to devise an image sharing platform for true-colour secret images.

Over the years, researches have incorporated data hiding, steganography and image authentication techniques into image sharing methods [24–31]. Data hiding and steganography techniques help conceal shadows in benign cover images that can go unnoticed by adversaries. After being reconstructed, the recovered image can be verified using image authentication techniques.

Recently, medical applications hugely reap the benefits of image sharing researches. Many medical image sharing schemes have been proposed [32–35]. In 2011, Ulutas et al. [35] proposed a sharing scheme for storing medical images and EPRs (electronic patients’ records) based on Shamir’s secret sharing scheme. Besides EPR hiding and confidentiality, this scheme also ensures the authenticity of the recovered images. In 2013, Fatma et al. [33] employed cryptographic encryption to securely share medical images in cloud storage, in which a system with three levels of security was provided to facilitate the communications between doctors and cloud service providers. Later, in 2014, Anbarasi and Mala [32] combined Shamir’s secret sharing with DNA cryptography to share medical images. They hid EPRs into medical images using DNA hiding techniques and used Huffman coding to compress the images before constructing shadows using Shamir’s technique. Also in 2014, Tso et al. [34] utilized a visual sharing technique to divide medical images into meaningless shares. One of the advantages of this scheme is its simplicity, stacking the shares immediately to reveal the original medical image.

3. Hamming Codes

Detecting transmission errors is highly significant in data communication. The simplest technique in detecting an error is to append a parity bit to each byte of the transmitted data. This mechanism detects the occurrence of error correctly if the number of error bits is an odd number. However, it cannot pinpoint the error bit by a byte.

Hamming code is superior to parity check because it can detect and identify the location of a single-bit error in the transmitted data [36]. Hamming codes are available in different sizes. If the number of parity bits is $m (\geq 3)$, then the code length is $n = 2^m - 1$. Hamming code has a minimum Hamming distance 3, and thus it can correct one single error. For example, for $m = 3$, the code length is 7 bits in which there are three parity bits and four data bits; this code is called (7,4) Hamming code. Similarly, (15,11) Hamming code has four parity bits and eleven data bits. The number of data bits increases exponentially as the number of parity bits grows.

The parity-check matrix H of Hamming code consists of all non-zero m -tuple 2^m as its columns (i.e., $2^m - 1$ columns). Suppose that the parity-check matrix is $H = [I_m Q]$, where I_m is an $m \times m$ identity matrix and Q is an m -tuple with weight two or more. Then the generator matrix is $G = [Q^T I_{n-m}]$, where $G \times H^T = [0]_{m \times (n-m)}$. In this paper, our approach adopts (7,4) Hamming code and (15,11) Hamming code. The following two examples show how to encode and decode for these two Hamming codes, respectively.

Example 1. For (7,4) Hamming code, H is a 3×7 matrix and G is a 4×7 matrix, as shown below. Obviously, one can easily verify $G \times H^T = [0]_{4 \times 3}$.

$$H = [I_3 Q] = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

$$G = [Q^T I_4] = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The above code is the so-called systematic code, i.e., the 4 data bits (d_1, d_2, d_3 and d_4) are at the most right 4 bits, and others are parity bits. To encode a 4-bit data $d = [d_1 d_2 d_3 d_4]$ into a 7-bit codeword c , the following equation is applied:

$$c = d \times G = [p_1 p_2 p_3 d_1 d_2 d_3 d_4]. \tag{1}$$

From Equation (1), we have $p_1 = d_1 \oplus d_3 \oplus d_4$, $p_2 = d_1 \oplus d_3 \oplus d_4$, and $p_3 = d_2 \oplus d_3 \oplus d_4$. It is observed that there are at least 2 parity bits covering a data bit, so even a parity bit is flipped we still can detect and correct it. For example, suppose that the data is $d = [0\ 1\ 0\ 1]$, then $p_1 = 1$, $p_2 = 1$ and $p_3 = 0$; the codeword is $c = [1\ 1\ 0\ 0\ 1\ 0\ 1]$. For decoding, the syndrome, a 3-tuple, is computed as

$$s = c' \times H^T = [s_3\ s_2\ s_1]. \tag{2}$$

$s = (s_3, s_2, s_1) = c' \times H^T$. If the syndrome is a zero vector $[0\ 0\ 0]$, the codeword is correct; otherwise, the syndrome $(s_3\ s_2\ s_1)$ can be used for correcting one error. For the received word $c' =$ characteristic $[1\ 1\ 1\ 0\ 1\ 0\ 1]$, the syndrome vector $[s_3\ s_2\ s_1]$ equals to $[0\ 0\ 1]$. The 3-tuple $(0\ 0\ 1)$ is the third row in H^T , and thus the bit p_3 (the position is 3 from left) should be flipped. The correct codeword is $[1\ 1\ 0\ 0\ 1\ 0\ 1]$.

Example 2. The (15,11) Hamming code is constructed in the same manner as the (7,4) Hamming code. The systematic forms of 4×15 parity-check matrix H and 11×15 generator matrix G are shown below.

$$H = [I_4Q] = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$G = [Q^T I_{11}] = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

An 11-bit data $d = [d_1\ d_2\ d_3\ d_4\ d_5\ d_6\ d_7\ d_8\ d_9\ d_{10}\ d_{11}]$ can be encoded into a systematic 15-bit codeword $c = d * G = [p_1\ p_2\ p_3\ p_4\ d_1\ d_2\ d_3\ d_4\ d_5\ d_6\ d_7\ d_8\ d_9\ d_{10}\ d_{11}]$, where the parity bits are computed as follows:

$$p_1 = d_1 \oplus d_2 \oplus d_4 \oplus d_7 \oplus d_8 \oplus d_9 \oplus d_{11},$$

$$p_2 = d_1 \oplus d_3 \oplus d_5 \oplus d_7 \oplus d_8 \oplus d_{10} \oplus d_{11},$$

$$p_3 = d_2 \oplus d_3 \oplus d_6 \oplus d_7 \oplus d_9 \oplus d_{10} \oplus d_{11},$$

$$\text{and } p_4 = d_4 \oplus d_5 \oplus d_6 \oplus d_8 \oplus d_9 \oplus d_{10} \oplus d_{11}.$$

For an 11-bit data $d = [0\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 1]$, the parity bits are $p_1 = 1$, $p_2 = 1$, $p_3 = 1$ and $p_4 = 0$. Thus, the codeword is $c = [1\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 1\ 0\ 1]$. Suppose that the received word is $c' = [1\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ 1]$. The syndrome of c' : $s = (s_4, s_3, s_2, s_1)$ is $(1\ 0\ 0\ 1)$. Then, we can locate the error bit which is at the eighth row in the matrix H^T (the error position is 8 from the left).

4. The Proposed Scheme

Suppose that there are two medical images (I_1 and I_2) for the same patient. The proposed scheme produces two shadows (S_1 and S_2) using the shadows generating algorithm. These two shadows have very little resemblance with the original images. If a malicious entity could, however, acquire one and only one shadow, he/she could not perceive any meaningful information about the patient's condition from it. For diagnosis purpose, having a shadow is as good as none.

When an authorized doctor would like to access this patient’s medical images, he/she must obtain both shadows S_1 and S_2 . The original image reconstruction algorithm is then used to regenerate the original medical images for diagnosis. Because two shadows can be used to reproduce the two original medical images, we have to put them on two different databases. This is a security measure ensuring that any security breach happened to one server will not compromise the whole system. Without the shadow stored on the other server, the patients’ health information does not leak out after the incidence. In the following sections, the shadow generation and original images reconstruction algorithms are described in detail using both (7,4) Hamming and (15,11) Hamming codes.

4.1. Using (7,4) Hamming Code

This shadows generation algorithm will produce two shadows (S_1 and S_2) from two medical images I_1 and I_2 . It manipulates the images at pixel level, therefore, one pixel from each original image will be picked sequentially for creating a new pair of pixels of the two shadows.

Suppose that pixel $P_1 = [i_{1,1} i_{1,2} i_{1,3} i_{1,4} i_{1,5} i_{1,6} i_{1,7} i_{1,8}]$ from I_1 and $P_2 = [i_{2,1} i_{2,2} i_{2,3} i_{2,4} i_{2,5} i_{2,6} i_{2,7} i_{2,8}]$ from I_2 are selected as shown in Figure 1, where $i_{1,1}, i_{1,2}, \dots, i_{1,8}$ and $i_{2,1}, i_{2,2}, \dots, i_{2,8}$ are the bits in those two pixels. Two corresponding pixels P'_1 and P'_2 on shadows S_1 and S_2 are produced as follows:

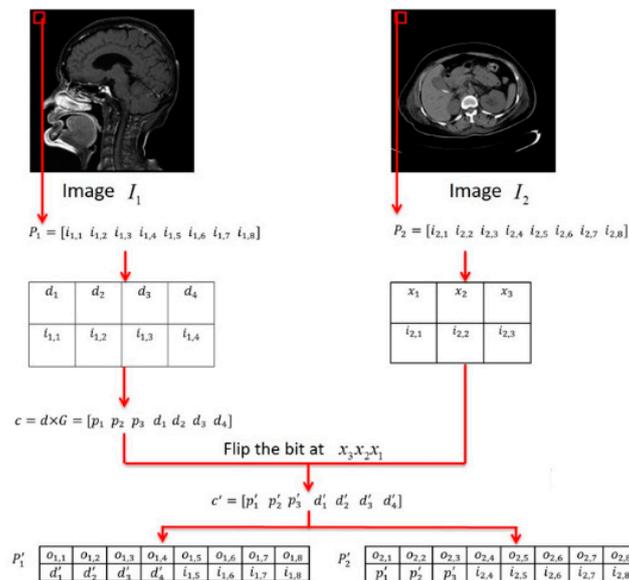


Figure 1. Shadow construction process using (7,4) Hamming code.

Algorithm: Shadow generation algorithm

Input: P_1 and P_2

Output: P'_1 and P'_2

Step 1. Assign $[d_1 d_2 d_3 d_4] = [i_{1,1} i_{1,2} i_{1,3} i_{1,4}]$, and $[x_1 x_2 x_3] = [i_{2,1} i_{2,2} i_{2,3}]$.

Step 2. Use (7, 4) Hamming code to compute $c = d * G = [p_1 p_2 p_3 d_1 d_2 d_3 d_4]$.

Step 3. For the codeword c , flip one bit at the position $(x_1 x_2 x_1)_2$, where x_1 is the least significant bit, to output a new codeword $c' = [p'_1 p'_2 p'_3 d'_1 d'_2 d'_3 d'_4]$.

Step 4. Construct two new pixels P'_1 and P'_2 on shadows as the following.

$$P'_1 = [o_{1,1} o_{1,2} o_{1,3} o_{1,4} o_{1,5} o_{1,6} o_{1,7} o_{1,8}]$$

$$= [d'_1 d'_2 d'_3 d'_4 i_{1,5} i_{1,6} i_{1,7} i_{1,8}],$$

and

$$P'_2 = [o_{2,1} o_{2,2} o_{2,3} o_{2,4} o_{2,5} o_{2,6} o_{2,7} o_{2,8}]$$

$$= [p'_1 p'_2 p'_3 i_{2,4} i_{2,5} i_{2,6} i_{2,7} i_{2,8}].$$

The above procedure runs through every pixel in the original images subsequently and produces the corresponding pixels for the shadows. Moreover, we alternately pick P_1 and P_2 (i.e., in the previous

round P_1 was selected from I_1 , then it will be obtained from I_2 in the next round) to make sure that the shadows are very much different from the original images.

Example 3. Suppose that two pixels selected from two medical images are $P_1 = [0\ 1\ 1\ 0\ 1\ 1\ 0\ 1]$ and $P_2 = [1\ 0\ 1\ 0\ 0\ 1\ 0\ 1]$. To generate two shadow pixels P'_1 and P'_2 , the following steps are executed:

Step 1. Determine d and x from P_1 and P_2 , $d = [0\ 1\ 1\ 0]$ and $x = [x_1\ x_2\ x_3] = [1\ 0\ 1]$.

Step 2. Encode d using (7, 4) Hamming code, we get

$$c = d * G = \begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}.$$

Step 3. Since $x = [1\ 0\ 1]$, flip the 5th bit (from left) of c to get $c' = [1\ 1\ 0\ 0\ 1\ 0\ 1]$.

Step 4. Since $d' = [0\ 0\ 1\ 0]$ and $p' = [1\ 1\ 0]$, two pixels $P'_1 = [0\ 0\ 1\ 0\ 1\ 1\ 0\ 1]$ and $P'_2 = [1\ 1\ 0\ 0\ 0\ 1\ 0\ 1]$ can be constructed.

We can see that the first four bits in P_1 have been changed from 0110 to 0010 in P'_1 . The pixel P'_2 has also changed since the first three bits have been changed from (101) to (110). As we mentioned earlier, P_1 and P_2 are selected alternately from the medical images I_1 and I_2 to ensure that changes are spread evenly between the two shadows images.

The above algorithm demonstrates how to create two shadow images from two medical images. The original image reconstruction algorithm shows how to reproduce the original images from the two shadows.

Suppose that pixels P'_1 and P'_2 are selected from shadows S_1 and S_2 in turn. The following steps are performed to reconstruct the original images.

Algorithm: Original images reconstruction algorithm

Input: P'_1 and P'_2

Output: P_1 and P_2

Step 1. Assign $[d'_1\ d'_2\ d'_3\ d'_4] = [o_{1,1}\ o_{1,2}\ o_{1,3}\ o_{1,4}]$ from first four bits of P'_1 , and $[p'_1\ p'_2\ p'_3] = [o_{2,1}\ o_{2,2}\ o_{2,3}]$ from first three bits of P'_2 .

Step 2. Construct the codeword $c' = [p'_1\ p'_2\ p'_3\ d'_1\ d'_2\ d'_3\ d'_4]$.

Step 3. Compute the syndrome $s = [s_3\ s_2\ s_1] = c' * H^T$.

Step 4. If s is not equal to $[0\ 0\ 0]$, go to Step 5; or else, stop the algorithm and return $P_1 = P'_1$ and $P_2 = P'_2$.

Step 5. Find the position of $(s_3\ s_2\ s_1)$ in H^T , i.e., $(x'_3\ x'_2\ x'_1)_2$ and then obtain the correct codeword $[p_1\ p_2\ p_3\ d_1\ d_2\ d_3\ d_4]$.

Step 6. Output the pixels for the original medical images as shown in Figure 2. The pixels P_1 and P_2 are reconstructed as follows:

$$P_1 = [i_{1,1}\ i_{1,2}\ i_{1,3}\ i_{1,4}\ i_{1,5}\ i_{1,6}\ i_{1,7}\ i_{1,8}]$$

$$= [d_1\ d_2\ d_3\ d_4\ o_{1,5}\ o_{1,6}\ o_{1,7}\ o_{1,8}],$$

and

$$P_2 = [i_{2,1}\ i_{2,2}\ i_{2,3}\ i_{2,4}\ i_{2,5}\ i_{2,6}\ i_{2,7}\ i_{2,8}]$$

$$= [x'_1\ x'_2\ x'_3\ o_{2,5}\ o_{2,6}\ o_{2,7}\ o_{2,8}].$$

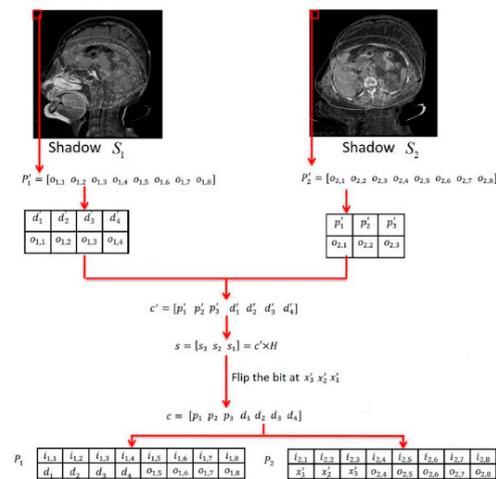


Figure 2. Original image reconstruction process using (7,4) Hamming code.

Example 4. In Example 3, it outputs two shadow pixels $P'_1 = [0\ 0\ 1\ 0\ 1\ 1\ 0\ 1]$ and $P'_2 = [1\ 1\ 0\ 0\ 0\ 1\ 0\ 1]$. Now we apply the original image reconstruction algorithm to get back the two original pixels P_1 and P_2 as follows.

- Step 1. Determine $d' = [0\ 0\ 1\ 0]$ and $p' = [1\ 1\ 0]$.
- Step 2. The codeword is $c' = [1\ 1\ 0\ 0\ 0\ 1\ 0]$.
- Step 3. Compute the syndrome of c' , we have

$$s = [s_3 s_2 s_1] = c' * H^T = [1\ 1\ 0\ 0\ 0\ 1\ 0] * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = [1\ 0\ 1]$$

Step 4. Since $s = [0\ 0\ 1]$ is not equal to $[0\ 0\ 0]$, go to Step 5.

Step 5. Flip the bit located at the position (1 0 1) in H^T (the fifth position from the left in c' , i.e., $(x'_3\ x'_2\ x'_1) = (101)$, and we have $c = [1\ 1\ 0\ 0\ 1\ 1\ 0]$.

Step 6. Obtain the original pixels $P_1 = [0\ 1\ 1\ 0\ 1\ 1\ 0\ 1]$ and $P_2 = [1\ 0\ 1\ 0\ 0\ 1\ 0\ 1]$.

It shows that the recovered pixels are correct; thus, the algorithms function correctly as expected.

So far, we have only demonstrated how to generate shadows and reconstruct original images, but we have not mentioned exactly which bits are selected to go through the processes. In a pixel, there are three groups of bits that we can select from (a) the least significant bits, (b) the most significant bits, and (c) those bits in the middle. As shown in the experiments, we will show how the selection of the group of bits for our algorithms affects the outcomes significantly.

In the following theorem, we theoretically prove that our scheme based on (7,4) Hamming code is a (2,2) secret sharing scheme satisfying the threshold condition, which only two shadows S_1 and S_2 can collaborate to recover the original images I_1 and I_2 . Meanwhile, any one shadow S_1 (respectively, S_2) cannot obtain the original image I_1 (respectively, I_2).

Theorem 1: The proposed scheme based on (7,4) Hamming code is a (2,2) secret sharing scheme.

Proof: To prove that the proposed scheme is a (2,2) secret sharing scheme, we have to prove the security condition (any one shadow cannot obtain its original image), and the threshold condition (two shadows can collaborate together to recover both original images).

We first prove the security condition that any one shadow S_j ($j = 1$ or 2) cannot obtain the original image I_j . Because P_1 and P_2 are selected alternately from medical images I_1 and I_2 (see Step 4 in the shadows generation algorithm), there are the following three cases.

Case 1: Shadow S_j has the pixel $[p'_1 p'_2 p'_3 i_{j,4} i_{j,5} i_{j,6} i_{j,7} i_{j,8}]$:

No matter the single error occurs in which bit of $[p_1 p_2 p_3 d_1 d_2 d_3 d_4]$, information of $(p'_1 p'_2 p'_3)$ is completely unrelated to information of $(x_1 x_2 x_3)$, i.e., the original $(i_{j,1} i_{j,2} i_{j,3})$. Also, in this case, we do not have $(d'_1 d'_2 d'_3 d'_4)$. Thus, we cannot obtain the correct codeword $[p_1 p_2 p_3 d_1 d_2 d_3 d_4]$ to further locate the position of error. So, the value of $(x_1 x_2 x_3)$ cannot be obtained and the pixel value is different from the original one.

Case 2: Shadow S_j has the pixel $[d'_1 d'_2 d'_3 d'_4 i_{j,5} i_{j,6} i_{j,7} i_{j,8}]$ and the error occurs in $(d'_1 d'_2 d'_3 d'_4)$:

The two 4-tuple $(d'_1 d'_2 d'_3 d'_4)$ and $(d_1 d_2 d_3 d_4)$ differ from one bit. For this case, there is no $(p'_1 p'_2 p'_3)$. Thus, the correct codeword $[p_1 p_2 p_3 d_1 d_2 d_3 d_4]$ cannot be obtained to locate the position of the error in $(d'_1 d'_2 d'_3 d'_4)$. So, we do not have the correct $(d_1 d_2 d_3 d_4)$ and thus the pixel value is different from the original one.

Case 3: Shadow S_j has the pixel $[d'_1 d'_2 d'_3 d'_4 i_{j,5} i_{j,6} i_{j,7} i_{j,8}]$ without the error in $(d'_1 d'_2 d'_3 d'_4)$:

These two 4-tuple $(d'_1 d'_2 d'_3 d'_4)$ and $(d_1 d_2 d_3 d_4)$ are the same, and thus the pixel value on shadow is the same as the original pixel.

The pixels P_1 and P_2 are selected alternately from the medical images I_1 and I_2 . There are 50% ($\bullet = 4/8$) probability for Case 1. The value of $(x_1 x_2 x_3)$ causes the error to occur in $(d'_1 d'_2 d'_3 d'_4)$ with 50% probability, and causes the error to occur in $(p'_1 p'_2 p'_3)$ and no error with 50% ($\bullet = 3/8 + 1/8$) probability. Therefore, all the pixels of the shadow S_j ($j = 1$ or 2) have probabilities of 50%, 25% and 25% for Case 1, Case 2 and Case 3, respectively. Finally, there are 75% of pixels different from the original pixels. Even though the other 25% of pixels have the same values as those of the original pixels (Case 3), one does not know the position of these pixels. From the above description, we cannot obtain the original image I_j from one S_j . The security condition is achieved.

Next, we prove the threshold condition. Suppose that we have both shadows S_1 and S_2 . From $[p'_1 p'_2 p'_3 d'_1 d'_2 d'_3 d'_4]$, the correct codeword $[p_1 p_2 p_3 d_1 d_2 d_3 d_4]$ is recovered, and thus the value of $(x_1 x_2 x_3)$ is obtained. Finally, the original pixels are recovered. \square

4.2. Using (15,11) Hamming Code

When using (15,11) Hamming code, the algorithms for shadow generation and original image reconstruction are similar to those in using (7,4) Hamming code. The difference is in the number of bits from the pixels of the original images selected for computing. Using (7,4) Hamming code we can select from the most significant bits, or from the least significant bits, or from the middle bits of a pixel, whereas we can only select either the most significant bits or the least significant bits of a pixel when using (15,11) Hamming code because we use at least 7 bits from a pixel in our algorithms. Algorithms of shadow generation and the recovery of original images are briefly described below.

Algorithm: Shadows generation algorithm

Input: P_1 and P_2

Output: P'_1 and P'_2

Step 1. Assign $[d_1 d_2 d_3 d_4 d_5 d_6 d_7 d_8 d_9 d_{10} d_{11}] = [i_{1,1} i_{1,2} i_{1,3} i_{1,4} i_{1,5} i_{1,6} i_{1,7} i_{1,8} i_{2,1} i_{2,2} i_{2,3}]$, and $[x_1 x_2 x_3 x_4] = [i_{2,4} i_{2,5} i_{2,6} i_{2,7}]$.

Step 2. Use (15,11) Hamming code to compute $c = d * G = [p_1 p_2 p_3 p_4 d_1 d_2 d_3 d_4 d_5 d_6 d_7 d_8 d_9 d_{10} d_{11}]$.

Step 3. For the codeword c , flip the bit at the position $(x_4 x_3 x_2 x_1)_2$ to output a new codeword $c' = [p'_1 p'_2 p'_3 p'_4 d'_1 d'_2 d'_3 d'_4 d'_5 d'_6 d'_7 d'_8 d'_9 d'_{10} d'_{11}]$.

Step 4. Construct two new pixels P'_1 and P'_2 on shadows as the following.

$$P'_1 = [o_{1,1} o_{1,2} o_{1,3} o_{1,4} o_{1,5} o_{1,6} o_{1,7} o_{1,8}]$$

$$= [d'_1 d'_2 d'_3 d'_4 d'_5 d'_6 d'_7 d'_8],$$

and

$$P'_2 = [o_{2,1} o_{2,2} o_{2,3} o_{2,4} o_{2,5} o_{2,6} o_{2,7} o_{2,8}]$$

$$= [d'_9 d'_{10} d'_{11} p'_1 p'_2 p'_3 p'_4 i_{2,8}].$$

Note: Similar to the algorithms using (7,4) Hamming code, the input pixels are taken alternatively from two original images. P_1 is first taken from I_1 , then in the next round P_1 is taken from I_2 ; the same goes for P_2 . This ensures that the outputs are mixed up nicely to produce the shadows.

Algorithm: Original image reconstruction algorithm

Input: P'_1 and P'_2

Output: P_1 and P_2

Step 1. Assign $[d'_1 d'_2 d'_3 d'_4 d'_5 d'_6 d'_7 d'_8 d'_9 d'_{10} d'_{11}] = [o_{1,1} o_{1,2} o_{1,3} o_{1,4} o_{1,5} o_{1,6} o_{1,7} o_{1,8} o_{2,1} o_{2,2} o_{2,3}]$ from all eight bits of P'_1 and the first 3 bits of P'_2 , and $[p'_1 p'_2 p'_3 p'_4] = [o_{2,4} o_{2,5} o_{2,6} o_{2,7}]$.

Step 2. Construct the codeword $c' = [p'_1 p'_2 p'_3 p'_4 d'_1 d'_2 d'_3 d'_4 d'_5 d'_6 d'_7 d'_8 d'_9 d'_{10} d'_{11}]$.

Step 3. Compute the syndrome $s = [s_4 s_3 s_2 s_1] = c' * H^T$.

Step 4. If s is not equal to $[0 0 0 0]$, go to Step 5; or else, stop the algorithm and return $P_1=P'_1$ and $P_2=P'_2$.

Step 5. Find the position of $(s_4 s_3 s_2 s_1)$ in H^T , i.e., $(x'_4 x'_3 x'_2 x'_1)_2$ and then obtain the correct codeword $[p_1 p_2 p_3 p_4 d_1 d_2 d_3 d_4 d_5 d_6 d_7 d_8 d_9 d_{10} d_{11}]$.

Step 6. Output the pixels for the original medical images. Pixels P_1 and P_2 are reconstructed as follows:

$$P_1 = [i_{1,1} i_{1,2} i_{1,3} i_{1,4} i_{1,5} i_{1,6} i_{1,7} i_{1,8}] \\ = [d_1 d_2 d_3 d_4 d_5 d_6 d_7 d_8],$$

and

$$P_2 = [i_{2,1} i_{2,2} i_{2,3} i_{2,4} i_{2,5} i_{2,6} i_{2,7} i_{2,8}] \\ = [d_9 d_{10} d_{11} x'_1 x'_2 x'_3 x'_4 o_{2,8}].$$

Theorem 2: The proposed scheme based on (15,11) Hamming code is a (2,2) secret sharing scheme.

Proof: We first prove the security condition that any one shadow S_j ($j= 1$ or 2) cannot obtain the original image I_j . Because P_1 and P_2 are selected alternately from medical images I_1 and I_2 (see Step 4 in Shadows generation algorithm), there are the following three cases.

Case 1: Shadow S_j has the pixel $[d'_9 d'_{10} d'_{11} p'_1 p'_2 p'_3 p'_4 i_{j,8}]$:

By a similar approach used in proving Theorem 1, the value of $(x_1 x_2 x_3 x_4)$ cannot be obtained. Thus, the pixel value is different from the original one.

Case 2: Shadow S_j has the pixel $[d'_1 d'_2 d'_3 d'_4 d'_5 d'_6 d'_7 d'_8]$ with one error:

By a similar approach in proving Theorem 1, we cannot obtain the correct codeword $[p_1 p_2 p_3 p_4 d_1 d_2 d_3 d_4 d_5 d_6 d_7 d_8 d_9 d_{10} d_{11}]$ to locate the position of error in $(d'_1 d'_2 d'_3 d'_4 d'_5 d'_6 d'_7 d'_8)$. So, the pixel values are different from the original one.

Case 3: Shadow S_j has the pixel $[d'_1 d'_2 d'_3 d'_4 d'_5 d'_6 d'_7 d'_8]$ without error:

These two pixels $(d'_1 d'_2 d'_3 d'_4 d'_5 d'_6 d'_7 d'_8)$ and $(d_1 d_2 d_3 d_4 d_5 d_6 d_7 d_8)$ are the same, and thus have the same pixel values.

Pixels P_1 and P_2 are selected alternately from medical images I_1 and I_2 . There is a 50% probability for Case 1. The value of $(x_1 x_2 x_3 x_4)$ causes the error to occur in $(d'_1 d'_2 d'_3 d'_4 d'_5 d'_6 d'_7 d'_8)$ with a 50% ($\therefore = 8/16$) probability, and meanwhile, the error occurs in $(d'_9 d'_{10} d'_{11} p'_1 p'_2 p'_3 p'_4)$ and no error has 50% ($\therefore = 7/16 + 1/16$) probability. All the pixels of the shadow S_j ($j = 1$ or 2) have 50%, 25% and 25% probabilities for Case 1, Case 2 and Case 3, respectively. Finally, there are 75% of pixels different from the original pixels. Even though the other 25% of pixels have the same values as those of the original pixels (Case 3), one does not know the position of these pixels. From the above description, we cannot obtain the original image I_j from one S_j . The security condition is achieved.

Next, we prove the threshold condition. Suppose that we have both shadows S_1 and S_2 . From $[p'_1 p'_2 p'_3 p'_4 d'_1 d'_2 d'_3 d'_4 d'_5 d'_6 d'_7 d'_8 d'_9 d'_{10} d'_{11}]$, the correct codeword $[p_1 p_2 p_3 p_4 d_1 d_2 d_3 d_4 d_5 d_6 d_7 d_8 d_9 d_{10} d_{11}]$ is recovered, and thus the value of $(x_1 x_2 x_3 x_4)$ is obtained. Finally, the original pixels are recovered. □

5. Experiments and Discussions

5.1. Experimental Results

In our experiments, first, four pairs of 138×138 medical images of scale X-ray scans were selected, which are the modality of computed tomography. In addition, all experiments were implemented by MATLAB R2017a, and the simulation environment for experiments was an Intel(R) Core(TM) i5-8500 v5 Hexa-core processor with 8 GB of RAM. The original images and their shadows are shown in Tables 1 and 2 for algorithms using (7,4) Hamming and (15,11) Hamming, respectively. The medical images in each pair were the scan images of different organs of one patient.

Table 1. Experimental result on medical images using (7,4) Hamming code.

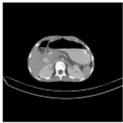
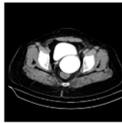
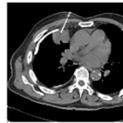
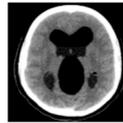
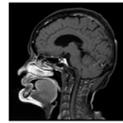
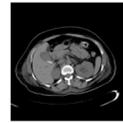
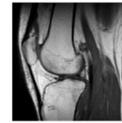
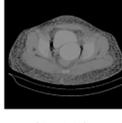
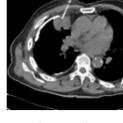
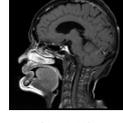
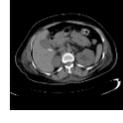
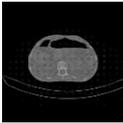
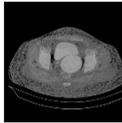
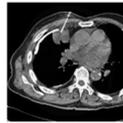
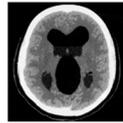
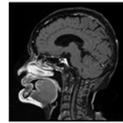
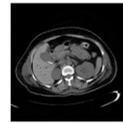
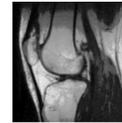
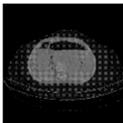
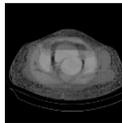
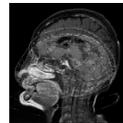
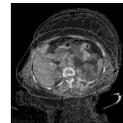
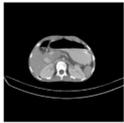
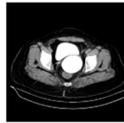
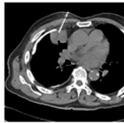
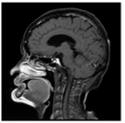
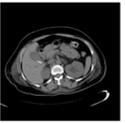
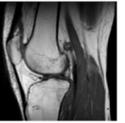
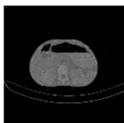
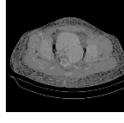
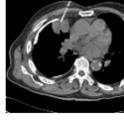
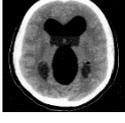
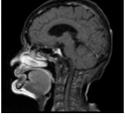
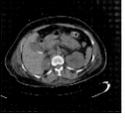
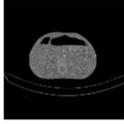
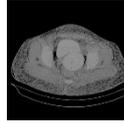
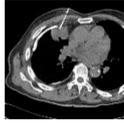
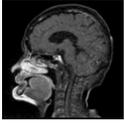
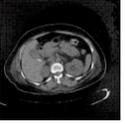
Comparison of Experiment Results							
Pair 1		Pair 2		Pair 3		Pair 4	
							
(1-1)	(1-2)	(1-3)	(1-4)	(1-5)	(1-6)	(1-7)	(1-8)
Original Medical Images							
Pair 1		Pair 2		Pair 3		Pair 4	
							
(1-9) PSNR = 44.14	(1-10) PSNR = 41.90	(1-11) PSNR = 40.09	(1-12) PSNR = 41.41	(1-13) PSNR = 42.24	(1-14) PSNR = 40.35	(1-15) PSNR = 40.79	(1-16) PSNR = 40.84
Shadow Images Produced by Using LSBs							
Pair 1		Pair 2		Pair 3		Pair 4	
							
(1-17) PSNR = 26.35	(1-18) PSNR = 25.49	(1-19) PSNR = 23.57	(1-20) PSNR = 24.19	(1-21) PSNR = 24.79	(1-22) PSNR = 24.23	(1-23) PSNR = 23.26	(1-24) PSNR = 24.42
Shadow Images Produced by Using Middle Bits							
Pair 1		Pair 2		Pair 3		Pair 4	
							
(1-25) PSNR = 18.97	(1-26) PSNR = 19.95	(1-27) PSNR = 14.10	(1-28) PSNR = 13.84	(1-29) PSNR = 16.66	(1-30) PSNR = 13.23	(1-31) PSNR = 13.38	(1-32) PSNR = 14.75
Shadow Images Produced by Using MSBs							

Table 2. Experimental result on medical images using (15,11) Hamming code.

Comparison of Experiment Results							
Pair 1		Pair 2		Pair 3		Pair 4	
							
(2-1)	(2-2)	(2-3)	(2-4)	(2-5)	(2-6)	(2-7)	(2-8)
Original Medical Images							
Pair 1		Pair 2		Pair 3		Pair 4	
							
(2-9) PSNR = 28.96	PSNR = 27.37	PSNR = 26.90	PSNR = 23.66	PSNR = 25.20	PSNR = 21.09	PSNR = 21.82	PSNR = 21.05
Shadow Images Produced by Using LSBs							
Pair 1		Pair 2		Pair 3		Pair 4	
							
PSNR = 26.84	PSNR = 27.55	PSNR = 22.14	PSNR = 21.52	PSNR = 21.99	PSNR = 17.63	PSNR = 19.14	PSNR = 19.32
Shadow images produced by using MSBs							

In (7,4) Hamming algorithm, up to 4 bits were manipulated in each pixel; thus, we could choose which group of bits to use, whether it is LSBs, MSBs, or in the middle of the pixels. For each pair of original images, three pairs of shadows were created. The first shadow pair was constructed using the least significant bits (LSBs) in each pixel. The second pair was produced using the middle bits of the pixels. And the last pair was generated using the most significant bits (MSBs).

(15,11) Hamming code did not generate the same choices as (7,4) Hamming. We used full 8 bits of a pixel from one original image and 7 bits of a pixel from the other image to construct shadows and reconstruct original images; thus, we were only allowed to choose the group of 7 bits either from LSBs or from MSBs.

The goal of our scheme was to produce shadows that look much different from the original ones; the less similarity between the original and shadow images, the better the result achieved. Even though the shadows resembled the original medical images, the details of those images, however, were smeared badly that would not be used in diagnosis or would leak patients' confidential health information.

To evaluate the results, we computed the peak signal-to-noise ratio (PSNRs) for each shadow. The higher the PSNR, the more similarity between the shadow and the original image. Therefore, the lower the PSNR the more secure our scheme is.

Table 1 shows that using (7,4) Hamming with LSBs yields PSNRs in the range from 40 dB to 45 dB. Manipulating the middle bits results in PSNRs from 23 dB to 27 dB. If we choose to use MSBs, the resulted PSNRs are between 13 dB and 19 dB, which is less than a third of the PSNRs if we choose to use LSBs, and a half of those if we choose to use the middle bits. Thus, using MSBs would produce more blurring effect compared with using another group of bits. Besides, it is worth to note that the size of the created shadows is the same as the original images in this paper.

The visual perception of looking at the three sets of shadow images clearly shows that manipulating the most significant bits of the X-rays scanned images has superior results compared with manipulating

the other groups of bits in pixels. The shadow images produced by using MSBs have their details blurred out thoroughly. Each shadow looks as if it is a result of the overlapping of the two original images. From one shadow it is obvious that we could not reconstruct the corresponding original image, let alone the other original.

Table 2 confirms that using MSBs gives a better result than using LSBs. Since we use 7 out of 8 bits in a pixel in (15,11) Hamming algorithms, the difference in PSNRs is subtle. Compared with (7, 4) Hamming algorithms, using (15,11) Hamming yields higher PSNRs in both LSBs and MSBs cases. Visually we can see that the result in (7,4) Hamming with MSBs is the best choice for our scheme.

To further demonstrate the visual effects provided by our schemes, the comparisons in specific features that have much difference between the original images and the corresponding shadows are depicted in Table 3, where the size of images is 512×512 . Among which, the former image is computed tomography and the other belongs to other modality. The more different the image is, the better the result achieves. As can be seen from Table 3, some of the outlines have blurred, especially in the experiments of pairs 2 and 4, where the shadow images are produced by using MSBs. It means that those images are badly damaged so that they would not be used in diagnosis. It is concluded that the proposed scheme does well in protecting the confidentiality of a patient's health.

The computation cost of performing the proposed algorithms is low due to the efficiency of computing Hamming code. Implementing Hamming code on the graphics processing unit (GPU) [30] can speed up the process 99 times compared with the normal sequential approach, and ensure a fast response to those applications that are time-sensitive. Thus, our proposed scheme can be implemented in such an efficient way and we can offer instant medical images for doctors, which is valuable considering the fact that doctors are known to be very busy when making rounds and visiting patients in the inpatient department. To prove this point better, the execution time of creating shadows and reconstructing original images corresponding to experiments in Table 1, are listed in Table 4. It can be observed from Table 4 that the average execution time of creating shadows and recovering the original image are around 0.6857s and 0.7447s, respectively. It is so fast that is suitable to respond to time-sensitive applications. It should be noted that all images used in Table 4 are 8-bit depth grey-scale images (each one is a single 2D slice) with the size of 512×512 .

Table 3. Comparisons of specific features between original images and shadow images using the proposed scheme.

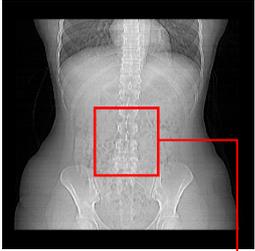
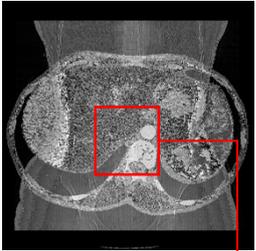
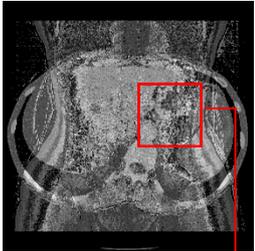
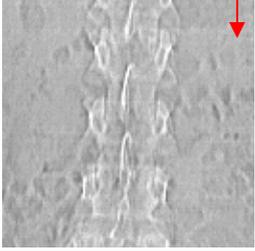
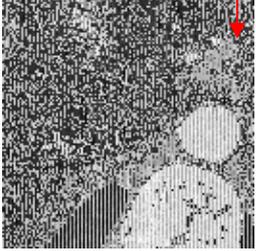
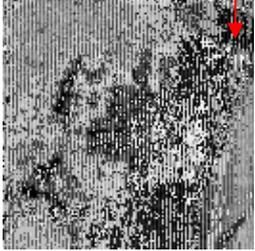
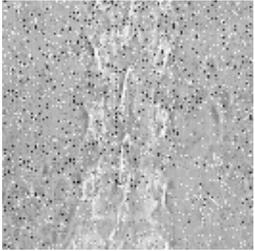
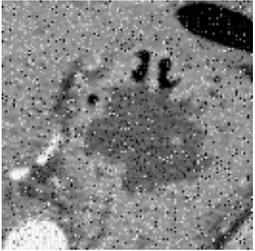
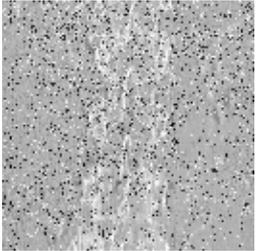
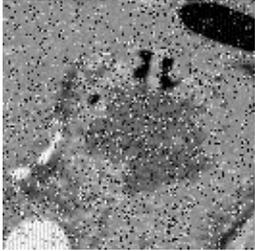
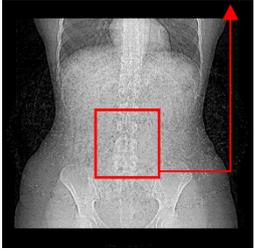
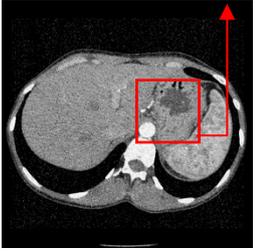
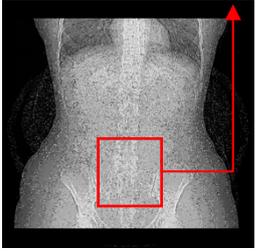
Original Medical Images		Shadow Images Produced by Using (7,4) Hamming Code (MSBs)	
Pair 1		Pair 2	
			
			
			
Pair 3		Pair 4	
			
Shadow images produced by using (15,11) Hamming code (LSBs)		Shadow images produced by using (15,11) Hamming code (MSBs)	

Table 4. Experimental results of execution time (s) for our proposed schemes.

Methods	Time (s)			
	(MSBs)	Creating Shadow (s)	Reconstructing Original Images (s)	Execution Time (s)
(7,4) Hamming code	Pair 1	0.6060	0.6104	1.2164
	Pair 2	0.6653	0.6711	1.3364
	Pair 3	0.6507	0.6576	1.3083
	Pair 4	0.6344	0.6908	1.3252
(15,11) Hamming code	Pair 1	0.7054	0.8147	1.5201
	Pair 2	0.7534	0.8403	1.5937
	Pair 3	0.6996	0.8011	1.5007
	Pair 4	0.7704	0.8715	1.6419
Average		0.6857	0.7447	1.4303

5.2. Comparisons

To further demonstrate the advantages of our proposed scheme, we compared the various features, including execution time, visual perception, non-expanded pixel, and lossless recovery, between our schemes and other related works [31,32,34,37]. As can be seen in Table 5, due to the efficiency of computing Hamming code, the execution time of our approach is relatively short, which means that it can be used in most real-time systems. At the same time, the image quality of shadows provided by ours is lower than other schemes [31,32,37], indicating that the visual perception of shadows is unable to support the basic conditions of the doctor’s diagnosis. And if the diagnosis is really required, the original images also can be recovered losslessly. In a sense, it protects the patient’s health information well shown.

Table 5. Comparison of various features for different schemes.

Features	Ref. [31]	Ref. [34]	Ref. [32]	Ref. [37]	Ours (MSBs)
Kernel	Magic matrix	Random grid	DNA cryptography	Polynomial	Hamming code
Average Execution Time (s)	1.38	0.18	-	-	0.69
Visual Perception (PSNR)	51.72/45.70	-	46.8–47	<30	19.15/18.47
Non-Expanded Pixel	Yes	Yes	No	Yes	Yes
Lossless Recovery	Yes	Yes	Yes	No	Yes

5.3. Discussions

Next, we discuss the following issues of our scheme more in detail: (1) compromised shadows (2) why use (7,11) and (15,11) Hamming codes in our scheme, and (3) why not just use a scrambling algorithm to get a less distorted shadow in our scheme.

Compromised shadow images:

The proposed scheme is a user-friendly like secret image sharing scheme [37–40], in which the shadow represents a distorted version of the original image. One can see a blurred medical image and can combine two shadows to reveal the original images. However, what will happen if shadows are compromised (e.g., attacked and compromised by geometric distortion, compression, filtering, and other image processing)? Because of the threshold condition, the original medical image will not be recovered when any one shadow is compromised. No measure can be taken to avoid this. Actually, this is not a disadvantage. It is to assure the security of a (2,2) scheme because our goal is to prevent the patients’ health information from being leaked via (2,2) a secret sharing scheme.

Why using (7,11) and (15,11) Hamming codes in our scheme:

BCH codes have the following parameters: block length $n = 2m - 1$, number of parity bits $(n - k) \leq mt$, and the minimum distance $d_{\min} = 2t + 1$, where m ($m \geq 3$) is a positive integer and t is the error correcting capability. Actually, BCH code is a generalization of the Hamming code. For $t = 1$, a BCH

code is the Hamming code with $n = 2m - 1$, $(n - k) = m$, and $d_{\min} = 3$. Our approach deals with two pixels P_1 and P_2 simultaneously. There are a total of 16 bits in the pair (P_1 and P_2). Thus, BCH codes with $m \geq 5$ having block length $n \geq 31$ larger than 16 cannot be used in our two-pixel approach. BCH code with $m = 3$ is the (7,4) Hamming code. For $m=4$, there are three BCH codes with $t = 1, 2$ and 3 , respectively. Note: for $t = 1$, it is the (15,11) Hamming code. On the other hand, for $t = 2$ and 3 , the codes are (15,7) BCH code and (15,5) BCH code.

Next, we describe why we do not use (15,7) BCH code and (15,5) BCH code. If we adopt these two codes in our scheme, we should flip at most two bits (there are $\binom{15}{0} + \binom{15}{1} + \binom{15}{2} = 121$ ways to determine the positions) for (15,7) BCH code., and at most three bits (there are $\binom{15}{0} + \binom{15}{1} + \binom{15}{2} + \binom{15}{3} = 576$ ways to determine the positions) for (15,5) BCH code. For these two cases, we need 7 ($\because 27 \geq 121$) bits and 10 bits ($\because 210 \geq 576$) to determine the positions to flip the bits. There are two weakness for such a process: one is complex, and the other is that the probability of choosing the way for flipping bit is unequal because $\binom{15}{0} + \binom{15}{1} + \binom{15}{2} \neq 2^7$ and $\binom{15}{0} + \binom{15}{1} + \binom{15}{2} + \binom{15}{3} \neq 2^{10}$. However, Hamming codes are perfect codes (i.e., $\binom{7}{0} + \binom{7}{1} = 2^3$ for (7, 4) Hamming code and for (15,11) Hamming code. Therefore, our approaches based on (7,4) and $\binom{15}{0} + \binom{15}{1} = 2^4$ for (15,11) Hamming codes have the same probability of choosing the way for flipping bit.

Why not just use a scrambling algorithm to get a less distorted shadow in our scheme:

Our goal is not just to let the shadow image look much different from the original one. The proposed scheme is a user-friendly (2,2) secret image sharing scheme. So, our scheme not only makes shadow different from the original one (i.e., providing user-friendly features) but also recovers the original medical image via two shadows (i.e., providing the threshold property of a (2,2) secret sharing scheme). However, any scrambling or related algorithm cannot achieve the same goal. There are many approaches to scramble bits in pixels. It should be carefully designed to achieve a threshold (2,2) scheme; rather than any scrambling tool can be easily applied on a (2,2) user-friendly secret sharing scheme. A well-known implementation of the secret sharing scheme is polynomial-based secret sharing. But the polynomial-based approach is more complex than our Hamming-code-based method. Because our scheme is a (2,2) secret scheme, one may adopt a bit-wise XOR operation. Careful design is required to achieve the threshold property of (2,2) scheme as well as to retain blurred versions of medical images on S_1 and S_2 .

6. Conclusions

In this paper, we proposed a novel method that can be used to construct and securely store shadows of medical images. The best results can be achieved by manipulating the most significant bits in pixels. The shadows are completely blurred out and it is impossible to obtain the important details of medical images from them. Experimental results demonstrate that using (7,4) Hamming code gives a more desirable blurring effect than using (15,11) Hamming. Hamming code employs simple and fast matrix operations. As a result, our scheme runs fast at low computational costs. Thus, the scheme is suitable for mobile devices like tablets and phablets, which are used prevalently in medical practice today. Our scheme currently is designed for a false safe database system with two databases. It would be sufficient for small size hospitals or clinics. However, we foresee that the scheme can be made scalable for systems with more than two databases. Therefore, in future, we would expand this scheme to accommodate an arbitrary number of original images and shadows.

Author Contributions: Data curation, C.-C.C. (Ching-Chun Chang), J.B. and H.-D.L.; Formal analysis, J.B. and H.-D.L.; Funding acquisition, C.-C.C. (Ching-Chun Chang); Investigation, C.-C.C. (Ching-Chun Chang) and C.-C.C. (Chih-Cheng Chen); Methodology, L.L. and C.-C.C. (Chih-Cheng Chen); Project administration, L.L., C.-C.C. (Chih-Cheng Chen) and T.-H.M.; Resources, C.-C.C. (Chih-Cheng Chen) and T.-H.M.; Software, C.-C.C. (Ching-Chun Chang); Visualization, H.-D.L. and Meen Teen-Hang; Writing—original draft, H.-D.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by National Natural Science Foundation of China (No. 61370218), and Public Welfare Technology and Industry Project of Zhejiang Provincial Science Technology Department (No. LGG19F020016).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chen, J.; Zhu, Z.; Fu, C.; Yu, H.; Zhang, L. A fast chaos-based image encryption scheme with a dynamic state variables selection mechanism. *Commun. Nonlinear Sci. Numer. Simul.* **2015**, *20*, 846–860. [[CrossRef](#)]
2. Dong, C. Asymmetric color image encryption scheme using discrete-time map and hash value. *Optik. Int. J. Light Electron. Opt.* **2015**, *126*, 2571–2575. [[CrossRef](#)]
3. Liu, H.; Kadir, A.; Gong, P. A fast color image encryption scheme using one-time S-Boxes based on complex chaotic system and random noise. *Opt. Commun.* **2015**, *338*, 340–347. [[CrossRef](#)]
4. Mannai, O.; Bechikh, R.; Hermassi, H.; Rhouma, R.; Belghith, S. A new image encryption scheme based on a simple first-order time-delay system with appropriate nonlinearity. *Nonlinear Dyn.* **2015**, *82*, 107–117. [[CrossRef](#)]
5. Bullock, A.; Dimond, R.; Webb, K.; Lovatt, J.; Hardyman, W.; Stacey, M. How a mobile app supports the learning and practice of newly qualified doctors in the UK: An intervention study. *BMC Med. Educ.* **2015**, *15*, 71. [[CrossRef](#)]
6. Zhang, K.; Yang, K.; Liang, X.; Su, Z.; Shen, X.; Luo, H.H. Security and privacy for mobile healthcare networks: From a quality of protection perspective. *IEEE Wirel. Commun.* **2015**, *22*, 104–112. [[CrossRef](#)]
7. Zhang, L.; Jung, T.; Liu, C.; Ding, X.; Li, X.Y.; Liu, Y. Pop: Privacy-preserving outsourced photo sharing and searching for mobile devices. In Proceedings of the International conference on distributed computing systems, Columbus, OH, USA, 29 June–2 July 2015; pp. 308–317.
8. Shao, J.; Lu, R.; Lin, X. Fine-grained data sharing in cloud computing for mobile devices. In Proceedings of the 2015 IEEE conference on computer communications (INFOCOM), Kowloon, Hong Kong, China, 26 April–1 May 2015; pp. 2677–2685.
9. Ibtihal, M.; Hassan, N. Homomorphic encryption as a service for outsourced images in mobile cloud computing environment. In *Cryptography: Breakthroughs in Research and Practice*; IGI Global: Hershey, PA, USA, 2020; pp. 316–330.
10. Shankar, T.N.; Sahoo, G.; Niranjana, S. Image encryption for mobile devices. In Proceedings of the International Conference in Communication Control and Computing Technologies, Ramanathapuram, India, 7–9 October 2010; pp. 612–616.
11. La Rosa, M.; Rabinovich, M.I.; Huerta, R.; Abarbanel, H.D.I.; Fortuna, L. Slow regularization through chaotic oscillation transfer in an unidirectional chain of Hindmarsh–Rose models. *Phys. Lett. A* **2000**, *266*, 88–93. [[CrossRef](#)]
12. Wang, F.; Ding, J.; Dai, Z.; Peng, Y. An application of mobile phone encryption based on Fibonacci structure of chaos. In Proceedings of the 2010 Second World Congress on Software Engineering, Wuhan, China, 19–20 December 2010; Volume 2, pp. 97–100.
13. Kim, C. Data hiding using improving hamming code. *J. Inst. Electron. Inf. Eng.* **2013**, *50*, 180–186. [[CrossRef](#)]
14. Kim, C.; Yang, C.N. Data hiding based on overlapped pixels using hamming code. *Multimed. Tools Appl.* **2014**, *75*, 1–13.
15. Wang, J.T.; Chang, Y.C.; Yu, S.S.; Yu, C.Y. Hamming Code Based Watermarking Scheme for 3D Model Verification. In Proceedings of the International Symposium on Computer, Consumer and Control, Taichung, Taiwan, China, 10–12 June 2014; pp. 1095–1098.
16. Blakley, G.R. Safeguarding cryptographic keys. In Proceedings of the National Computer Conference, New York, NY, USA, 4–7 June 1979; Volume 48, pp. 313–317.
17. Shamir, A. How to share a secret. *Commun. ACM* **1979**, *22*, 612–613. [[CrossRef](#)]

18. Thien, C.C.; Lin, J.C. Secret image sharing. *Comput. Graph.* **2002**, *26*, 765–770. [[CrossRef](#)]
19. Wang, R.Z.; Su, C.H. Secret image sharing with smaller shadow images. *Pattern Recognit. Lett.* **2006**, *27*, 551–555. [[CrossRef](#)]
20. Wang, R.Z.; Shyu, S.J. Scalable secret image sharing. *Signal Process. Image Commun.* **2007**, *22*, 363–373. [[CrossRef](#)]
21. Chang, C.C.; Lin, C.C.; Lin, C.H.; Chen, Y.H. A novel secret image sharing scheme in color images using small shadow images. *Inf. Sci.* **2008**, *178*, 2433–2447. [[CrossRef](#)]
22. Chang, C.C.; Wu, M.N. An algorithm for color image compression base on common bit map block truncation coding. In Proceedings of the 6th Joint Conference on Information Science, Research Triangle Park, NC, USA, 8–13 March 2002; pp. 964–967.
23. Tsai, D.S.; Horng, G.; Chen, T.H.; Huang, Y.T. A novel secret image sharing scheme for true-color images with size constraint. *Inf. Sci.* **2009**, *179*, 3247–3254. [[CrossRef](#)]
24. Chang, C.C.; Lin, C.Y.; Tseng, C.S. Secret image hiding and sharing based on the (t, n)-threshold. *Fundam. Inform.* **2007**, *76*, 399–411.
25. Chang, C.C.; Chen, Y.H.; Wang, H.C. Meaningful secret sharing technique with authentication and remedy abilities. *Inf. Sci.* **2011**, *181*, 3073–3084. [[CrossRef](#)]
26. Eslami, Z.; Razzaghi, S.H.; Ahmadabadi, J.Z. Secret image sharing based on cellular automata and steganography. *Pattern Recognit.* **2010**, *43*, 397–404. [[CrossRef](#)]
27. Eslami, Z.; Ahmadabadi, J.Z. Secret image sharing with authentication-chaining and dynamic embedding. *J. Syst. Softw.* **2011**, *84*, 803–809. [[CrossRef](#)]
28. Lin, P.Y.; Chan, C.S. Invertible secret image sharing with steganography. *Pattern Recognit. Lett.* **2010**, *31*, 1887–1893. [[CrossRef](#)]
29. Wu, C.C.; Kao, S.J.; Hwang, M.S. A high quality image sharing with steganography and adaptive authentication scheme. *J. Syst. Softw.* **2011**, *84*, 2196–2207. [[CrossRef](#)]
30. Wu, X.; Ou, D.; Ling, Q.; Sun, W. A user-friendly secret image sharing scheme with reversible steganography based on cellular automata. *J. Syst. Softw.* **2012**, *85*, 1852–1863. [[CrossRef](#)]
31. Liu, Y.; Chang, C.C. A Turtle Shell-Based Visual Secret Sharing Scheme with Reversibility and Authentication. *Multimed. Tools Appl.* **2018**, *77*, 25295–25310. [[CrossRef](#)]
32. Anbarasi, L.J.; Mala, A. EPR hidden medical image secret sharing using DNA cryptography. *Int. J. Eng. Technol.* **2014**, *6*, 1346–1356.
33. Fatma, E.; Hikal, N.A.; Abou-Chadi, F.E.Z. Secret medical image sharing and EPR data embedding scheme over cloud computing environment. *Int. J. Comput. Appl.* **2013**, *69*, 19–26.
34. Tso, H.K.; Lo, T.M.; Chen, W.K. Friendly medical image sharing scheme. *J. Inf. Hiding Multimed. Signal Process.* **2014**, *5*, 367–378.
35. Ulutas, M.; Ulutas, G.; Nabyev, V.V. Medical image security and EPR hiding using Shamir’s secret sharing scheme. *J. Syst. Softw.* **2011**, *84*, 341–353. [[CrossRef](#)]
36. Morelos-Zaragoza, R.H. *The Art of Error Correcting Coding*, 2nd ed.; John Wiley & Sons: Hoboken, NJ, USA, 2006.
37. Yang, C.N.; Yu, K.H.; Lukac, R. User-friendly image sharing using polynomials with different primes. *Int. J. Imaging Syst. Technol.* **2007**, *17*, 40–47. [[CrossRef](#)]
38. Thien, C.C.; Lin, J.C. An image-sharing method with user-friendly shadow images. *IEEE Trans. Circuits Syst.* **2003**, *13*, 1161–1169. [[CrossRef](#)]
39. Islam, M.S.; Kim, C.H.; Kim, J.M. Computationally efficient implementation of a hamming code decoder using graphics processing unit. *J. Commun. Netw.* **2014**, *17*, 198–202. [[CrossRef](#)]
40. Vo, H.P. User-friendly sharing system using polynomials with different primes in two images. *Int. J. Comput. Appl.* **2014**, *86*, 40–45.

