Complex & Intelligent Systems (2023) 9:2699–2711
https://doi.org/10.1007/s40747-021-00391-0

ORIGINAL ARTICLE

Security for eHealth system: data hiding in AMBTC compressed images via gradient-based coding

Yung-Yao Chen[1] · Yu-Chen Hu[2] · Hsiang-Yun Kao[3] · Yu-Hsiu Lin[3]

Received: 9 November 2020 / Accepted: 29 April 2021 / Published online: 8 May 2021

Abstract
Various eHealth applications based on the Internet of Things (IoT) contain a considerable number of medical images and visual electronic health records, which are transmitted through the Internet everyday. Information forensics thus becomes a critical issue. This paper presents a data hiding algorithm for absolute moment block truncation coding (AMBTC) images, wherein secret data, or the authentication code, can be embedded in images to enhance security. Moreover, in view of the importance of transmission efficiency in IoT, image compression is widely used in Internet-based applications. To cope with this challenge, we present a novel compression method named gradient-based (GB) compression, which is compatible with AMBTC compression. Therefore, after applying the block classification scheme, GB compression and data hiding can be performed jointly for blocks with strong gradient effects, and AMBTC compression and data hiding can be performed jointly for the remaining blocks. From the experimental results, we demonstrate that the proposed method outperforms other state-of-the-art methods.

Keywords Visual health record security · Security for eHealth system · Absolute moment block truncation coding (AMBTC) · Data hiding

Introduction

Nowadays, people spend much more time on the Internet. With information becoming increasingly accessible and dispersed through the Internet and the trend of using the Internet for digital secret data transmission, the need to transmit physical files such as fingerprints and confidential design blueprints is decreasing. However, because the open Internet network is not a safe environment, information security methods such as data hiding are essential for the Internet of Things (IoT) and visual IoT.

✉ Yu-Hsiu Lin
yhlin@mail.ntut.edu.tw

[1] Department of Electronic and Computer Engineering, National Taiwan University of Science and Technology, Taipei 106, Taiwan (R.O.C.)

[2] Department of Computer Science and Information Management, Providence University, Taichung 433, Taiwan (R.O.C.)

[3] Graduate Institute of Automation Technology, National Taipei University of Technology, 1, Sec. 3, Zhongxiao E. Rd., Taipei 10608, Taiwan (R.O.C.)

In visual IoT, medical images or visual health records are common visual sensing data related to IoT-based e-health systems. As depicted in Fig. 1, numerous medical images are stored and transmitted in e-health systems, which are the privacy of each patient and cannot be leaked or tampered. Because these visual health records are assembled into big data, they require protection to prevent data from being obtained by hackers and personal information from being leaked during transmission, and also to ensure that personal information is not distributed maliciously [1,2]. Therefore, data hiding techniques have become more and more essential to increase the security of the IoT-based e-health systems.

Because compulsorily hiding unknown data into a cover image will reduce the image quality, data hiding methods can be further classified into the types of reversible data hiding and irreversible data hiding. Reversible data hiding methods [3–8] embed secret data into the image, and original image can be recovered from the marked image (i.e., the data-embedded image) during the decoding process. By contrast, irreversible data hiding methods [9–15] cannot reverse the marked image back to the cover image once data hiding process is completed. However, reversible data hiding methods normally require extra memory space to store the necessary

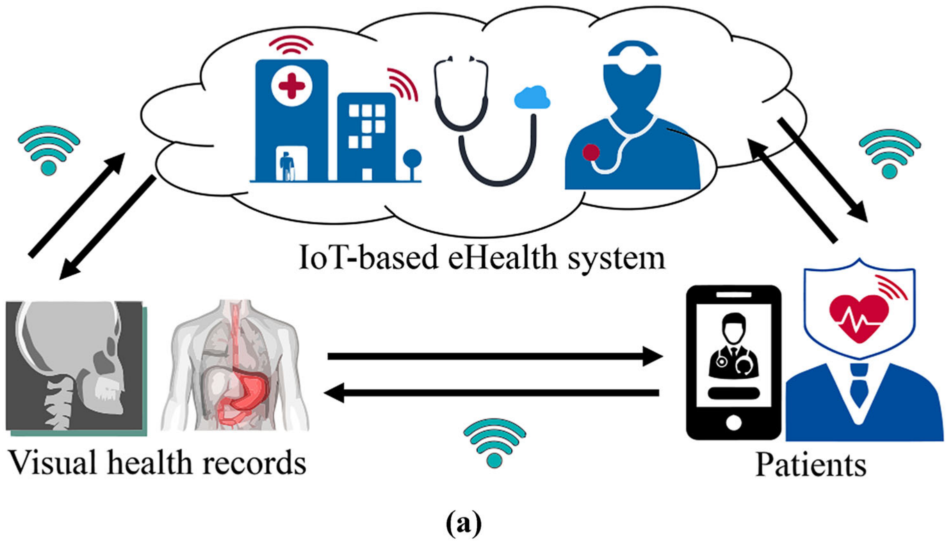


Fig. 1 Application scenario of the proposed work: transmission of visual electronic health records with enhanced security

data for restoration, and the decoding process is complicated so as to extract the secret data and to restore the original cover image simultaneously. Compared with reversible data hiding methods, irreversible data hiding methods have its own advantage of larger payload and faster decoding speed. Most irreversible data hiding methods pursue obtaining larger payload while keeping minimal image distortion or loss. Least Significant Bit (LSB), Exploiting Modification Direction (EMD), and Pixel Value Difference (PVD) are the three representative algorithms in irreversible data hiding. In LSB method, the secret data are embedded by changing the last few significant bits of the binary pixel values, so that the error can be significantly reduced. In EMD method [16], the pixels are distributed into several groups, and the secret data are embedded by modifying a single pixel of the corresponding pixel group each time. In PVD method [17], the secret data are embedded by manipulating the difference between the two pixel values, so that the image distortion can be diffused to local adjacent regions.

In this study, our priority is in high payload, fast calculation, and efficient transmission; therefore, the proposed method is an irreversible data hiding method in the compression domain. Among different image compression methods, the absolute moment block truncation coding (AMBTC) algorithm [18] is selected, where the cover image is divided into non-overlapping blocks of the same size (the details of AMBTC is described in the section "AMBTC compression"). In AMBTC, two values (high mean/low mean) are calculated to represent the information of each original image block. Therefore, AMBTC compression is suitable to the images with two sets of pixel characteristics, e.g., the images in which the edge or the contour is prominent. Many data hiding methods involving AMBTC compression have been recently proposed [19–23].

Block classification might be one of the most typical strategies used in AMBTC-based data hiding method. Ou and Sun [24] proposed an AMBTC-based data hiding method, which divides all image blocks into two types: smooth/complex blocks, by comparing the difference between high mean (HM) and low mean (LM) of a block with a pre-defined threshold. Subsequently, two different data hiding schemes are presented to each type of block, respectively (the details of [24] is described in "Ou and Sun's method"). Later, Malik et al. [25] inherited the block classification scheme of [24], but enhanced the data hiding method in the complex blocks. In [25], HM and LM are, respectively, replaced by four quantization levels which are stored by a 2-bit plane during the compression process. The secret data are embedded into the 2-bit plane to increase payload. Although [25] embeds the hidden data in the complex blocks, it modifies the compressed content of the complex block to maintain the image quality. Similarly, on the basis of block classification in [24], Hong [26] proposed an adaptive pixel pair matching (APPM) technique to enhance the data hiding method in the smooth blocks. Compared with [24], the method of [26] can embed more secret data in the set of selected HM and LM pairs using the APPM technique.

Different from the block classification scheme of [24], Kumar et al. [27] and Chang et al. [28] consider classifying the image blocks into three types: high-complex blocks, low-complex blocks, and smooth blocks. This kind of classification requires one more threshold to distinguish high-complex blocks from low-complex blocks, and thus, a more robust data-embedding strategy can be applied to the complex blocks. Normally, researchers take advantage of the property of the low-complex blocks to increase the payload. For example, the method of [27] embeds extra eight-bit secret data in the bitmap of each low-complex block through the hamming distance judgment method, and the method of [28] embeds extra six-bit secret data in each low-complex block by the matrix coding scheme.

Although the AMBTC method performs well for images in which edge characteristics are obvious, it is unsuitable for compressing a block with high-gradient characteristics. Gradation characteristic refers to distributions of pixels that are approximately equal to each other, and these approximately equal distributions have characteristics of light and shadows. AMBTC compression has difficulty retaining gradient characteristics; moreover, it causes block artifacts. In naturally shot images, as long as the image contains a light source, it inevitably produces a large amount of gradation. Thus, we aimed to create a gradient block compression method that emphasizes the characteristics of the gradient.

The motivation of this study was to create an adaptive compression method suited to blocks with gradation characteristics in the image, such that each block can maintain its corresponding characteristics and have a larger payload, better image quality, and less block effect. Our method takes advantage of features of the AMBTC method by dividing the image into two types of blocks, gradient blocks and complex blocks; each type is then divided into two categories. By referring to the data hiding method in [25], we changed the content of the image compression data appropriately to meet the need to hide data. However, compared with the minor corrections in [25], we changed the gradient blocks completely to our new gradient compression method to store block content.

The rest of this paper is organized as follows. In the section "Related works", the related works are reviewed. The section "Proposed method" presents the proposed method and its corresponding data extraction scheme. The section "Experimental results" provides the experimental results and the comparisons between the proposed method and other state-of-the-art methods. Finally, the conclusions are presented in the section "Conclusion".

Related works

AMBTC compression

This section introduces the basic idea of AMBTC compression algorithm. In the beginning, the original image has to be divided into non-overlapping 4×4 blocks. The compression is achieved by simplifying all the pixel values of each block into 2 quantification levels.

As shown in Fig. 2, the AMBTC algorithm can be viewed as a spatial-domain compression method. After dividing the original image into blocks, the pixel values in a block are quantized into two levels (named, high mean value and low mean value) by thresholding the block mean. For each block, the block mean (M) can be calculated by

$$M = \frac{1}{16}\sum_{i=1}^{16} x_i , \tag{1}$$

where x_i indicates the ith pixel value of the block. In addition, the two quantization levels (high mean HM and low mean LM) can be calculated by

$$\mathrm{HM} = \frac{1}{16-q}\sum_{x_i \geq M} x_i , \tag{2}$$

and

$$\mathrm{LM} = \frac{1}{q}\sum_{x_i < M} x_i , \tag{3}$$

where M indicates the block mean gray level, and q indicates the number of pixels whose values are less than M. The threshold value of AMBTC is set as the block mean to preserve the statistical moments of each image block. Therefore, AMBTC can preserve the absolute central moment of each original image block and can preserve the image quality after compression.

After comparing the pixel value of the corresponding position with M, the ith bit map B_i is generated, as shown in Fig. 2b. If the pixel value is less than M, it is set to be 0; otherwise, it is set to be 1. Therefore, the compression ratio of the AMBTC algorithm is 4—the memory storage required for the original block is 128 bits, but the memory storage required for the corresponding AMBTC block is only 32 bits. That is, two quantization levels (16 bits) and one bit map (16 bits) will be used to reconstruct the image block.

Ou and Sun's method [24]

Ou and Sun's method invented an AMBTC-based data hiding scheme, which embeds secret data by re-arranging the order of the bit stream and the bitmap in AMBTC. The main feature of this method is to divide all AMBTC blocks into two categories: smooth blocks and complex blocks. The classification is based on the comparison of the difference between HM − LM and a threshold thr. When HM − LM is less than thr, the block is considered as a smooth block, and the 16-bit secret data are embedded by replacing the bitmap of each smooth block completely. After the secret data are embedded, because the bitmap is changed, the method of [24] further modifies the high mean and low mean as

$$\mathrm{HM}' = \frac{1}{q_1'}\sum_{p_i \in B_1} x_i , \tag{4}$$

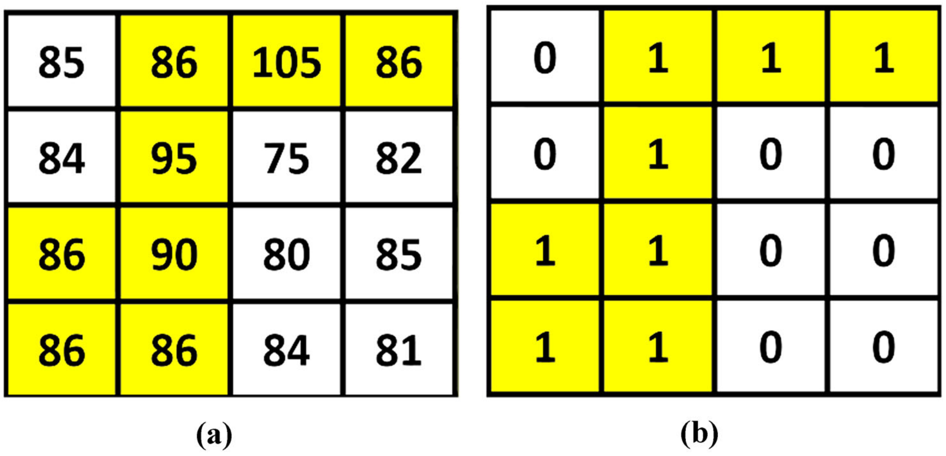


Fig. 2 One example of the AMBTC compression algorithm. **a** Input image block. **b** Output bitmap in which the quantization levels are LM = 82 and HM = 90

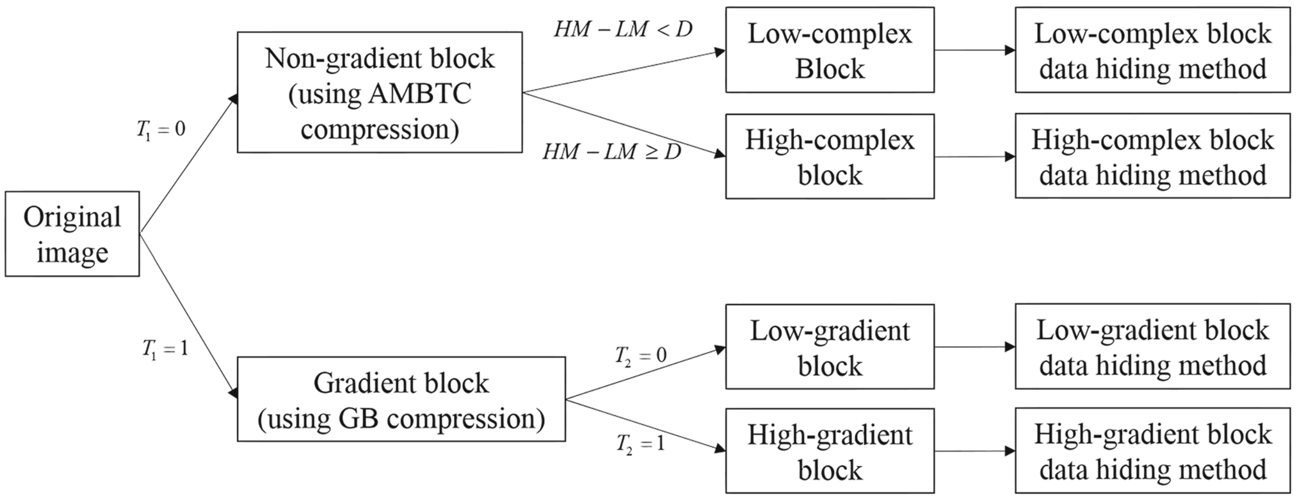

Original image
$T_1 = 0$
$T_1 = 1$
Non-gradient block (using AMBTC compression)
Gradient block (using GB compression)
$HM - LM < D$
$HM - LM \geq D$
$T_2 = 0$
$T_2 = 1$
Low-complex Block
High-complex block
Low-gradient block
High-gradient block
Low-complex block data hiding method
High-complex block data hiding method
Low-gradient block data hiding method
High-gradient block data hiding method

Fig. 3 Overall framework of the data hiding scheme

and

$$\mathrm{LM}' = \frac{1}{q_0'} \sum_{p_i \in B_0} x_i , \tag{5}$$

where the symbols q_1' and q_0', respectively, represent the number of 1 and 0 in the new bitmap. B_0 represents the position set, where the pixel value equals 0 in the new bitmap; and B_1 represents the position set, where the pixel value equals 1 in the new bitmap

On the other hand, each of the complex blocks is embedded into 1-bit secret data by simultaneously reversing the bit plane and exchanging the order of HM and HM in the compression trio code. That is, if the secret code is 0, the original compression trio code (HM, LM, B) is maintained. By contrast, if the secret code is 1, the trio code becomes (LM, HM, $\bar{B}$), where $\bar{B}$ denotes the reversed bit plane. Each of the smooth blocks is embedded into 16-bit secret data by completely replacing the bit plane. The two quantization levels are updated according to the embedded secret data. In our method, referring to the compression method of AMBTC, some blocks are compressed with AMBTC, and all the blocks are classified into different types of blocks for hiding data.

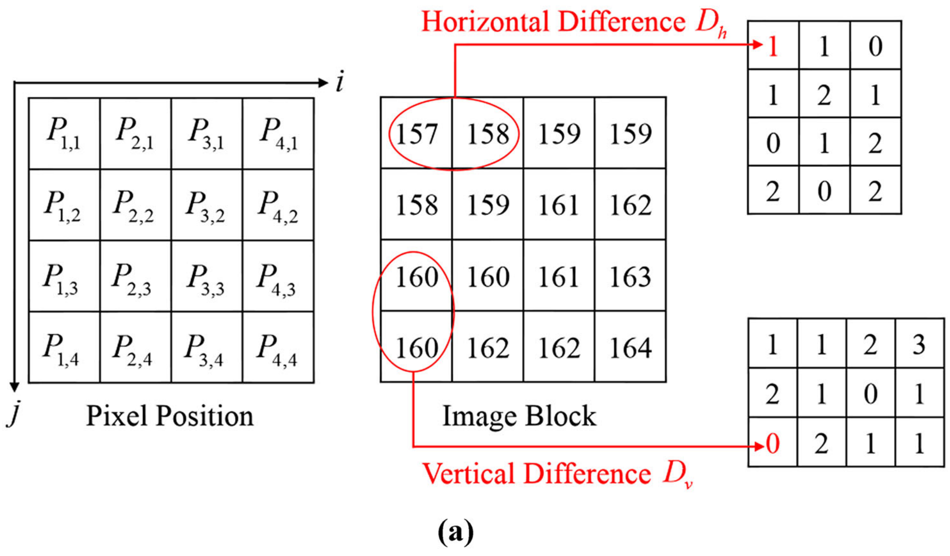
Proposed method

Figure 3 provides the framework of the proposed method. Similarly to conventional AMBTC, the input grayscale image is first divided into non-overlapped 4 × 4 image blocks. All image blocks are classified into one of four types; we designed different hiding schemes according to the local image properties.

Block classification

In naturally shot (not human-made) images, most image blocks contain gradient effects to various extents. This is because during the image-capturing process, light and shadow characteristics produce gradual shades of varying

Fig. 4 Example of calculating horizontal and vertical differences



luminance. In view of this property, we present a novel compression method named gradient-based (GB) compression, which has two advantages: (1) it is compatible with AMBTC compression; that is, for a 4 × 4 grayscale image block, the results of both GB and AMBTC compression are 32 bit-streams; and (2) for image blocks with strong gradient effects, by accompanying our proposed data hiding method, GB compression can create a higher hiding capacity and retain better image quality than AMBTC can.

To determine whether a block contains sufficient gradient property, we first define the horizontal difference (D_h) matrix and the vertical difference (D_v) matrix. As shown in Fig. 4, $P_{i,j}$ indicates the grayscale pixel values of an image block, and the horizontal/vertical difference can be, respectively, expressed as

$$D_h(i,j) = P_{i+1,j} - P_{i,j}, i \in 1,2,3, j \in 1,2,3,4, \quad (6)$$

and

$$D_v(i,j) = P_{i,j+1} - P_{i,j}, i \in 1,2,3,4, j \in 1,2,3. \quad (7)$$

In addition, the gradient consistency (G_c) index is defined as the total error obtained using the Pythagorean theorem with the root-mean-square error of D_h and D_v, as follows:

$$G_c = \sqrt{\mathrm{MSE}(D_h) + \mathrm{MSE}(D_v)}, \quad (8)$$

where MSE is the mean square error. As the value of G_c becomes smaller, the distribution of horizontal/vertical differences becomes more consistent over the image block, indicating the smoothness of the block.

This paper proposes a hierarchical strategy to classify the image blocks using three thresholds. In the first hierarchy, the threshold T_1 is used to distinguish gradient blocks from non-gradient blocks, and is defined as

$$T_1 = (G_c < \mathrm{thr})\,\&\,(|\bar{D}_h| < 16)\,\&\,(|\bar{D}_v| < 16), \quad (9)$$

where $\bar{D}_h$ and $\bar{D}_v$ are the absolute mean values of D_h and D_v, respectively. T_1 actually represents a Boolean function; when $T_1 = 1$, the block is defined as a gradient block and is followed by the proposed GB compression. When $T_1 = 0$, the block is defined as a non-gradient block and is followed by the traditional AMBTC compression. In the second hierarchy, all the gradient blocks are further classified into high-gradation blocks and low-gradation blocks using the threshold T_2:

$$T_2 = (G_c < \mathrm{thr})\,\&\,(|\bar{D}_h| < 4)\,\&\,(|\bar{D}_v| < 4). \quad (10)$$

As with T_1, the threshold T_2 is also a Boolean value, but it has stricter (4 versus 16, which are two empirical values) constraints in $\bar{D}_h$ and $\bar{D}_v$. On the other hand, all non-gradient (complex) blocks are further classified as high-complexity or low-complexity blocks by comparing the difference between the high mean and the low mean of an AMBTC block with the threshold D. Figure 5 illustrates an example of block classification.

To be able to extract the secret data, it is necessary to construct a table which can represent the type of all the blocks. Each bit in the table represents one of the blocks in cover image, so that all blocks are divided into gradient and non-gradient to record in this classification table. When the block is a gradient, the corresponding table position is recorded as 1, otherwise, recorded as 0.

Joint GB compression and data hiding in gradient blocks

In (9) and (10), if G_c, $\bar{D}_h$, and $\bar{D}_v$ are all small, the shades of intensity variation become imperceptible, indicating the existence of the gradient property. Therefore, we propose a joint GB compression and data hiding method that fully utilizes the gradient property, which is described as follows.

Figure 6a shows the output bit stream of the proposed method, where the first bit (red bit in Fig. 6a) is an identifier used to distinguish between high-gradient blocks and low-gradient blocks: for the case that the block is identified as a high-gradient block, it is recorded as 1; otherwise, it is recorded as 0. Subsequently, an 8-bit code (orange bits in Fig. 6a) is used to save the mean grayscale (M) of the original image block. If a block satisfies the Boolean function T_2 (i.e., a high-gradient block), the gradient effects of the horizontal/vertical direction are significantly prominent, and the pixel values are similar all over the block. Noticing that $\bar{D}_h < 4$ holds as $T_2 = 1$, we could estimate the horizontal gradient (G_h) by rounding $\bar{D}_h$ to the one of nearest eight quantization levels

$$G_h = \text{round}(\bar{D}_h) \text{ and } G_h \in \{-3.5, -2.5, \ldots, 3.5\}. \quad (11)$$

Therefore, a 3-bit code (dark green bits in Fig. 6a) is used to save the information of estimated horizontal gradient. Similarly, because $\bar{D}_v < 4$ holds as $T_2 = 1$, a 3-bit code (light green bits in Fig. 6a) is used to save the information of estimated vertical gradient (G_v)

$$G_v = \text{round}(\bar{D}_v) \text{ and } G_v \in \{-3.5, -2.5, \ldots, 3.5\}. \quad (12)$$

Finally, despite requiring 32 bits (same as using AMBTC compression) to save the information of this stego block, we create a 17-bit space for embedding the hidden data.

Figure 6b illustrates the reconstruction of the proposed method. To restore the grayscale image block, this work proposes utilizing the estimated gradients through a bilinear interpolation scheme

$$\text{RC}(i, j) = \text{round}[M+(i-2.5)\times G_h+(j-2.5)\times G_v], \quad (13)$$

where $\text{RC}(i, j), i, j \in 1, 2, 3, 4$ is the reconstructed pixel value of the (i, j) position. Different from AMBTC compression that uses high mean and low mean values, the proposed GB compression only requires a mean value to approximate the intensity information of the original block. In (13), the underlying concept of the reconstruction is based on the existence of a virtual center with grayscale, and the gradient property results in a two-dimensional arithmetic sequence in the reconstructed image block. Similar to (11) and (12), for the case that the block is identified as a low-gradient block, we use two 5-bit codes (representing the value from one of $\{-15.5, -14.5, \ldots, 14.5, 15.5\}$) to estimate vertical and horizontal gradients, respectively.

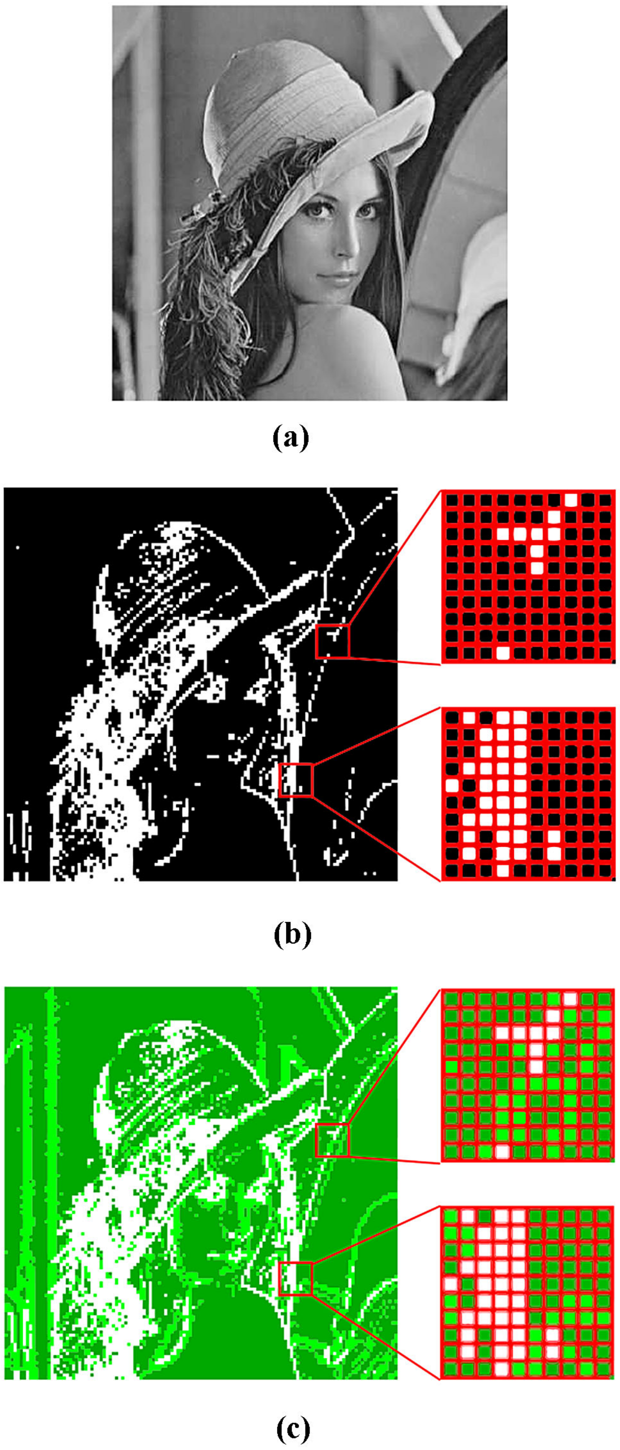


Fig. 5 Illustration of the proposed block classification scheme: **a** input grayscale image; **b** result after the 1st-hierarchy of block classification (thr = 12), where black and white units, respectively, represent the gradient and non-gradient blocks; and **c** result after the 2nd hierarchy of block classification, where dark green and light green indicate the high-gradient and low-gradient blocks, respectively

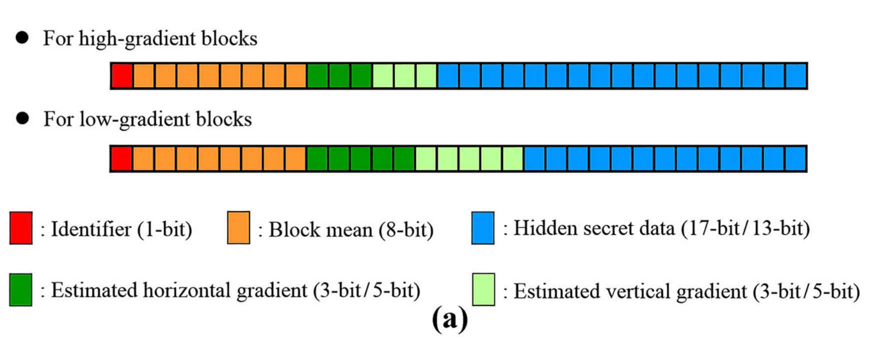
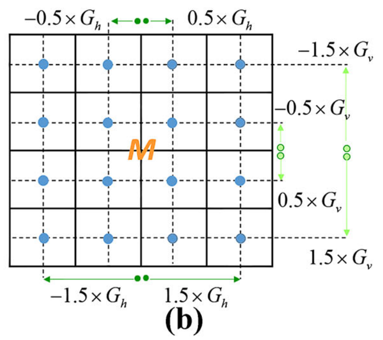

For high-gradient blocks
For low-gradient blocks
: Identifier (1-bit)
: Block mean (8-bit)
: Hidden secret data (17-bit/13-bit)
: Estimated horizontal gradient (3-bit/5-bit)
: Estimated vertical gradient (3-bit/5-bit)
(a)


Fig. 6 Illustration of the proposed method. **a** Jointly GB compression and data hiding. **b** Reconstruction of the gradient blocks

In the decoding process, after obtaining the bit stream of the gradient block, the identifier of the bit stream is used to determine whether the block is high-gradient block or low-gradient block. If the identifier is 1, the block is a high-gradient block, and the hidden data can be obtained by extracting from the 16th bit to the 32th bit. Otherwise, if the identifier is 0, the block is a low-gradient block, and the hidden data can be obtained by extracting from the 20th bit to the 32th bit.

Data hiding in non-gradient blocks

Here, we introduce the data hiding method for non-gradient blocks (i.e., complex blocks). First, we use the AMBTC method to compress the complex blocks to obtain the two quantization values HM, LM, and a bitmap B. The difference $d = \text{HM} - \text{LM}$ is used to distinguish between low-complexity blocks and high-complexity blocks.

The classification of high-complexity blocks and low-complexity blocks is shown in the right half of Fig. 3, where $D \in [4, 255]$. When $d > D$, the complex block is determined to be a high-complexity block; otherwise, the complex block is determined to be a low-complexity block. For the low-complexity blocks, we use the least significant bit data hiding method to hide 2-bit secret data in HM and LM. Let $S_1, S_2, S_3, S_4 \in \{0, 1\}$ denote the 4-bit secret data:

$$\begin{cases} S_H = S_1 \times 2 + S_2 \\ S_L = S_3 \times 2 + S_4 \end{cases}, \tag{14}$$

where $0 \leq S_H < 4$ and $0 \leq S_L < 4$ indicate the secret data embedded by modifying HM and LM, respectively. Then, the process of hiding data in HM and LM can be described as

$$\begin{cases} \text{HM}' = \text{HM} - \text{mod}(\text{HM} - S_H, 4), \text{ if HM} \geq 4 \\ \text{HM}' = S_H, \text{ if HM} < 4 \\ \text{lM}' = \text{LM} - \text{mod}(\text{LM} - S_L, 4), \text{ if LM} \geq 4 \\ \text{lM}' = S_L, \text{ if LM} < 4 \end{cases}, \tag{15}$$

and

$$\begin{cases} \text{LM}' = \text{lM}' + 4, \text{ if HM}' - \text{LM}' > D \\ \text{LM}' = \text{lM}', \text{ if HM}' - \text{LM}' \leq D \end{cases}, \tag{16}$$

where HM′ and LM′, respectively, indicate the modified mean values (HM and LM) that have been data embedded. In (15) and (16), S_H is embedded in HM, and S_L is embedded in LM.

In the high-complex blocks, we use Ou and Sun's method to hide 1-bit secret data as follows. When the secret data are code 1, we swap the location of HM and LM in the bit stream, and replace B with $\bar{B}$ which means the complement of B. If the secret data are code 0, we do nothing to the block. In this way, we complete the data hiding in the high-complex blocks.

Experimental results

In this section, we discuss the comparison charts of the peak signal-to-noise ratio (PSNR) corresponding to the payload generated by different thresholds using the proposed method as well as the results of comparing our method with other methods. We used two standard grayscale test images (Airplane and Lena, size of 512 × 512) and six medical images (Brainix, Cerebrix, Goudurix, Manix, Phenix, and Vix, size of 400 × 400) selected from the public dataset [29], as shown in Fig. 7.

For comparison with other methods, we selected PSNR, payload, and efficiency as evidence. PSNR represents the quality of the image. The higher the PSNR, the closer the pixel value of the marked image is to the original grayscale image. Payload represents how many secret data can be embedded in a test image. Efficiency represents the ratio of the payload to the total bits of the final data. The definition of PSNR can be described as

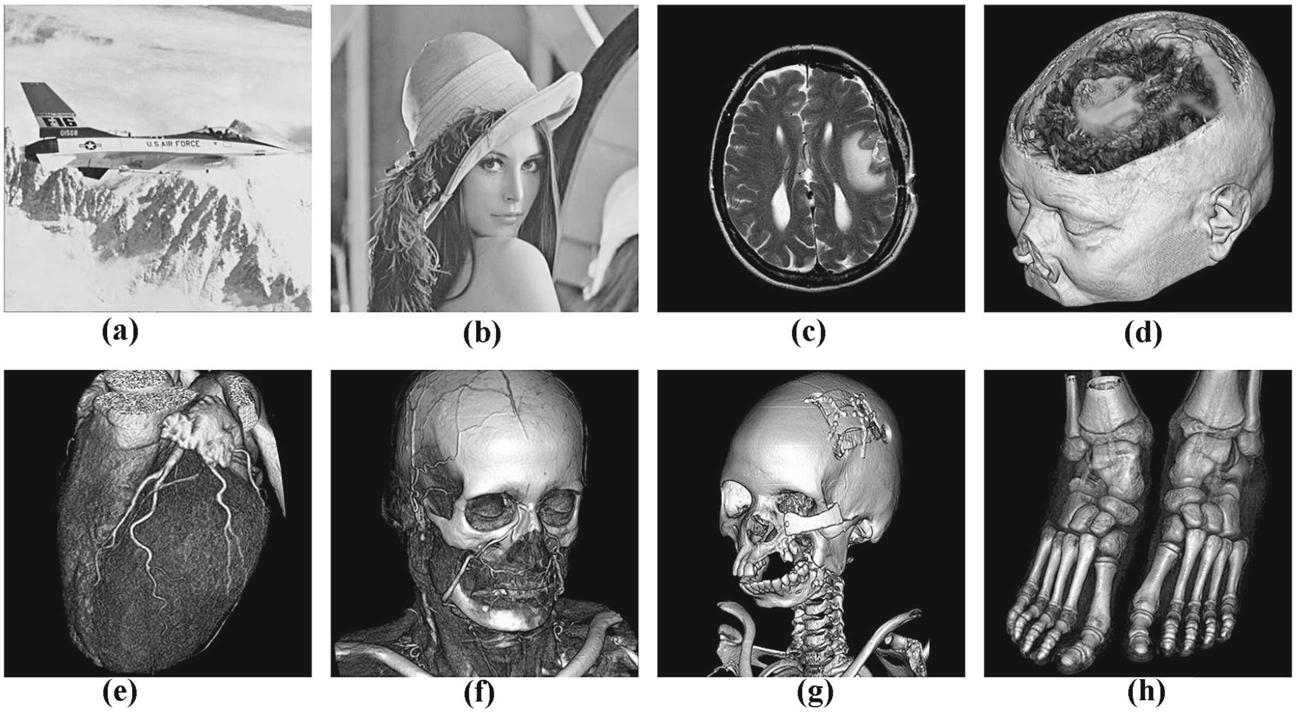


Fig. 7 Test images. **a** Airplane. **b** Lena. **c** Brainix. **d** Cerebrix. **e** Goudurix. **f** Manix. **g** Phenix. **h** Vix

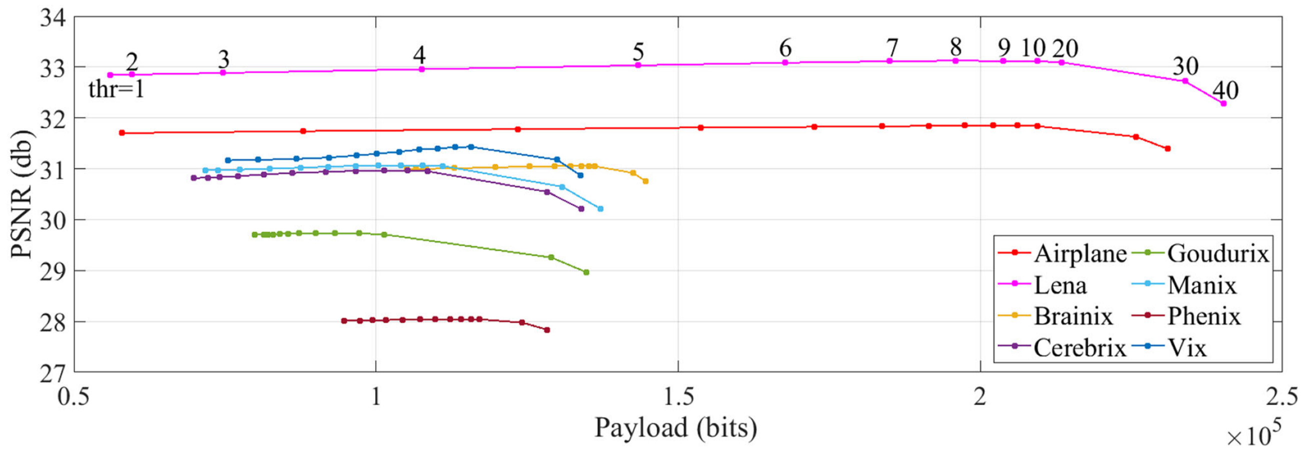


Fig. 8 When D is set as 25, the results of marked images as thr ranges from 1 to 40

$$10 \times \log_{10}\left(\frac{W \times H \times 255^2}{\sum_{W,H}\left[\sum_{m,n} g_{i+m,j+n} - h_{i+m,j+n}\right]^2}\right), \quad (17)$$

and the definitions of Efficiency can be described as

$$\text{Efficiency} = \frac{\text{Payload}}{C_{\text{total}}}, \quad (18)$$

where W and H represent the image width and the image height, respectively. The variable g represents the original image, and g represents the output stego AMBTC image. The PSNR value is the ratio between the maximum possible power of a signal and that of the corrupting noise, which is commonly used to measure the image quality. As PSNR value goes higher, it indicates the better quality of the stego images. In (18), C_{total} indicates a particular (fixed) length of the final code. Therefore, the efficiency value must be as high as possible.

Figure 8 demonstrates the influence of varying threshold (thr) values in our method. Let us set the value of D to 25

Table 1 Comparison between setting different thr values

Image	thr = 6				thr = 10			
	Gradient block (%)	Payload	PSNR	Efficiency	% of Gradient block (%)	Payload	PSNR	Efficiency
Airplane	62.25	186,828	31.83	34.55	74.69	209,086	31.85	38.67
Lena	54.96	170,092	33.24	31.46	76.48	211,575	33.11	39.13
Brainix	71.72	126,964	31.07	38.47	80.66	136,810	31.05	41.46
Cerebrix	44.52	88,409	30.92	26.79	63.22	107,323	30.96	32.52
Goudurix	42.96	87,896	29.73	26.64	53.72	99,463	29.73	30.14
Manix	48.67	94,344	31.04	28.59	62.92	109,795	31.06	33.27
Phenix	60.84	109,929	28.04	33.31	68.50	118,348	28.04	35.86
Vix	56.07	102,662	31.29	31.11	68.21	115,369	31.42	34.96

Table 2 Comparison of different methods in terms of PSNR, payload, and efficiency

Method	Metric	Test image							
		Airplane	Lena	Brainix	Cerebrix	Goudurix	Manix	Phenix	Vix
[24]	PSNR	31.71	32.72	30.95	30.83	29.75	30.97	28.01	31.17
	Payload	174,754	168,844	117,580	74,950	75,400	82,000	101,635	88,390
	Efficiency	33.33	32.20	36.74	23.42	23.56	25.63	31.76	27.62
[27]	PSNR	31.33	32.21	30.68	29.98	29.02	30.19	27.82	30.48
	Payload	200,912	200,514	130,250	101,608	100,808	106,328	117,118	109,234
	Efficiency	38.32	38.25	40.70	31.75	31.50	33.23	36.60	34.14
[28]	PSNR	29.94	30.88	29.59	28.73	27.84	29.02	26.92	29.40
	Payload	210,142	210,004	134,698	110,668	110,122	114,513	123,444	116,316
	Efficiency	40.08	40.06	42.09	34.58	34.41	35.79	38.58	36.35
Ours	PSNR	31.85	33.11	31.05	30.96	29.73	31.06	28.04	31.42
	Payload	209,086	211,575	136,810	107,323	99,463	109,795	118,348	115,369
	Efficiency	38.67	39.13	41.46	32.52	30.14	33.27	35.86	34.96

and observe the impact of thr on the PSNR and Payload as it increases from 1 to 40, as depicted in Fig. 8. When thr increases from 1 to 6, the PSNR increases slowly or remains constant. This occurs, because according to (9), thr affects the number of blocks that are determined to be gradient blocks; the number of gradient blocks increases with thr. Moreover, compared with complex blocks, gradient blocks usually have fewer pixel value errors. Thus, the PSNR value increases with the number of gradient blocks. In addition, as thr increases from 1 to 6, Payload increases rapidly, because the data capacity in a high-gradation block (or a low-gradation block) is 17 bits (or 13 bits), which is much higher than the data capacity of a complex block.

However, when thr is within the range of (6, 10], the overall PSNR might start to decrease. This happens, because when thr is larger than 6, we relax the error of determination as a gradient block, and the gradient characteristics of the resulting gradient block start to decrease. Furthermore, because the pixel error of the complex block is large, the PSNR increases when the number of complex blocks decreases. Therefore, as these two effects offset each other, the overall PSNR would have a similar range. When thr is larger than 10, the overall PSNR begins to decrease rapidly due to the decrease of the gradient characteristics in the gradient blocks. As evident in Fig. 8, thr has an optimal range. To optimize the PSNR result, we can set thr equal to 6. To achieve the maximum payload without a substantial loss in PSNR, we can set thr to $6 < \text{thr} \leq 10$.

Table 1 presents the results when we fix D to 25 and set thr to 6 or 10 to observe the proportion of gradient blocks occupied in eight test images. For the same image, when the proportion of gradient blocks increases, both Payload and Efficiency increase, which indicates that gradient blocks hide data more effectively than complex blocks do. Moreover, in the interval of thr $\in$ (6, 10], the PSNR does not excessively increase or decrease. Therefore, to reach the largest data capacity, we set thr = 10 in the subsequent experiments.

Subsequently, we compared our method with similar methods [24,27,28]. These three methods are similar to the proposed method in that the difference between HM and LM

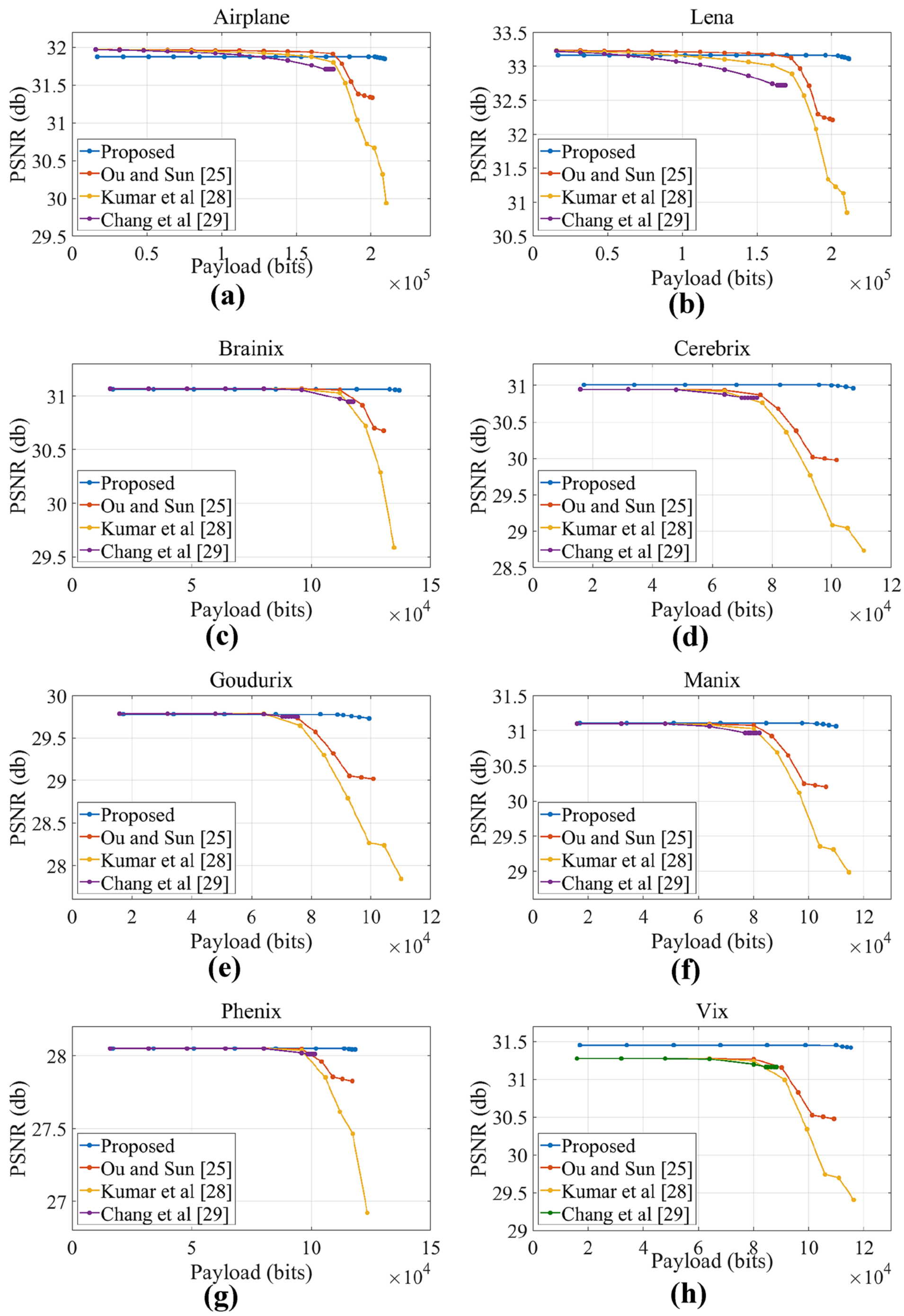


Fig. 9 Comparison of PSNR versus Payload among [24,27,28] and the proposed method

is used to distinguish complex blocks from smooth blocks. We used the fixed thresholds of $D = 25$ and thr = 10 as the uniform standards. The comparison results are listed in Table 2.

Figure 9 depicts the comparison of PSNR versus Payload. Our method obviously has a higher PSNR when the number of hidden data (i.e., Payload) is higher. One reason is that numerous blocks with gradient characteristics are set as gradient blocks to ensure that these blocks have a higher PSNR. Therefore, even if the payload increases, the decrease of PSNR is limited due to the smaller number of complex blocks. In other methods, when the number of hidden data becomes too large, complex blocks produce larger errors, which severely reduce the PSNR. Therefore, in our method, the quality of the marked image improves with the number of hidden data as the gradient characteristics are fully utilized. Finally, two examples of visual comparison are provided in Figs. 10 and 11. Figure 10 demonstrates that the proposed

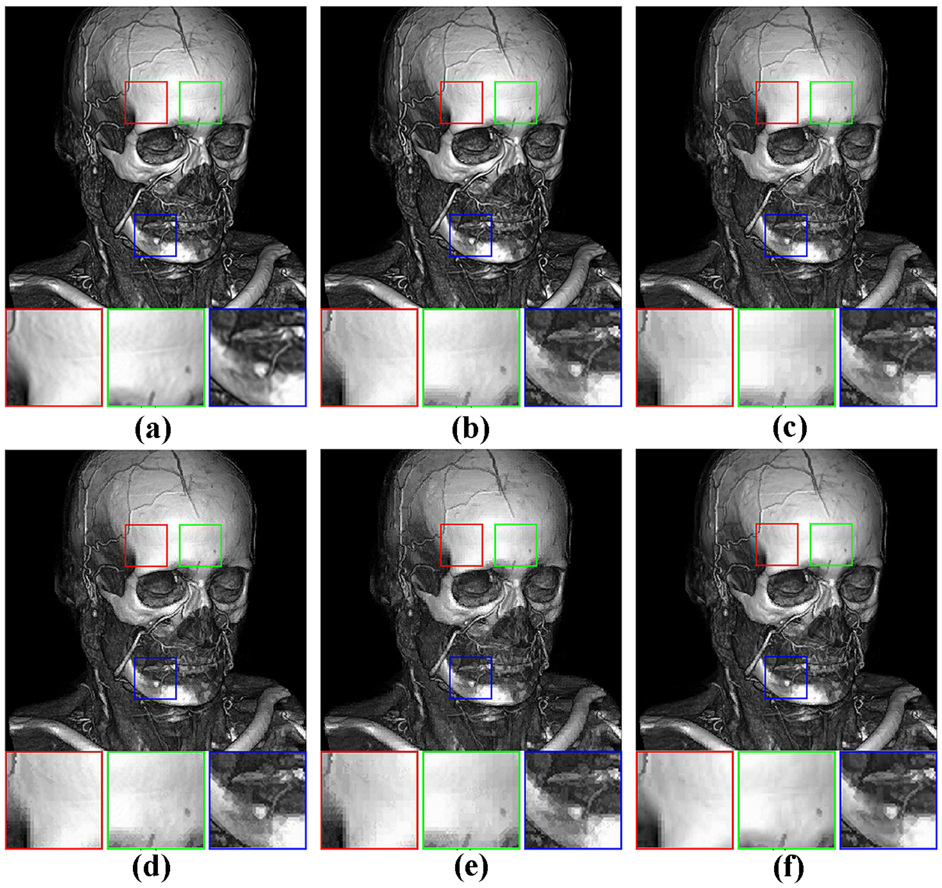


Fig. 10 Results of visual comparison using the test image Manix. **a** Original grayscale. **b** Result of AMBTC. **c** Result of [24]. **d** Result of [27]. **e** Result of [28]. **f** Result of the proposed method. From **c**–**f**, the threshold value is set as 8

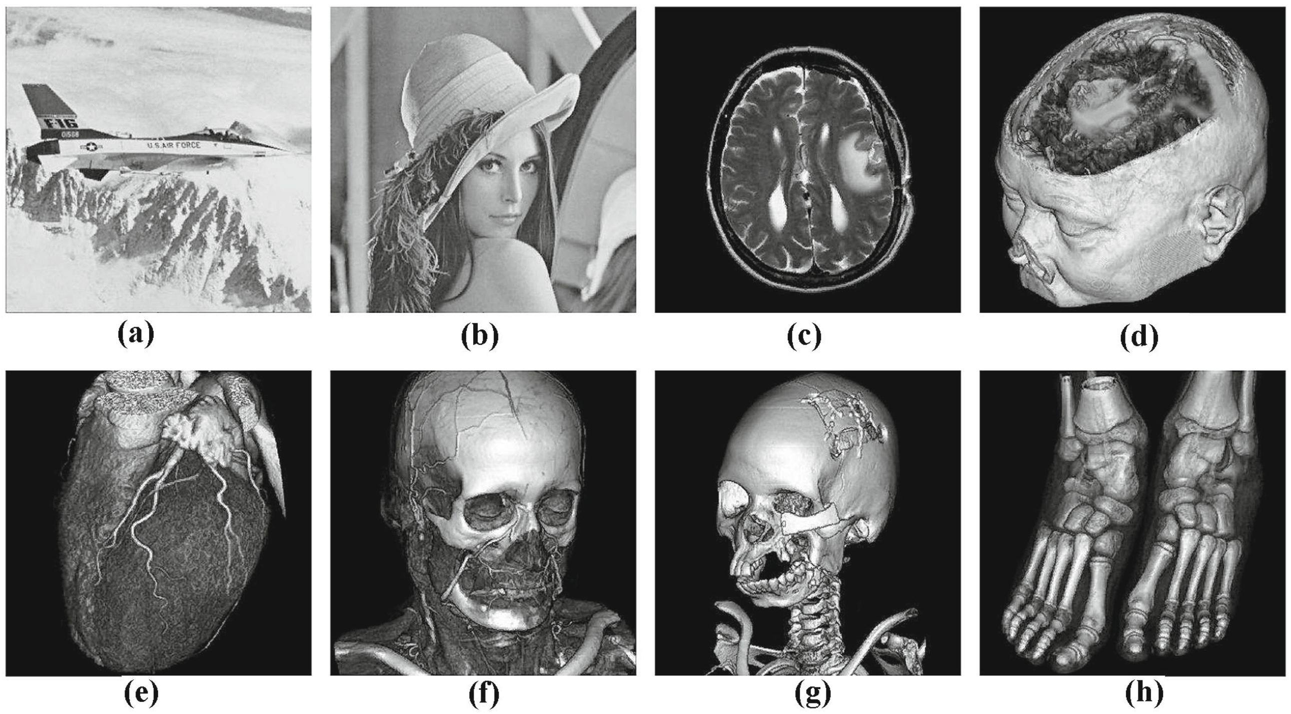


Fig. 11 Output stego images using the proposed method. **a** Airplane (Payload = 209,086 bits, PSNR = 31.85 dB). **b** Lena (Payload = 211,575 bits, PSNR = 33.11 dB). **c** Brainix (Payload = 136,810 bits, PSNR = 31.05 dB). **d** Cerebrix (Payload = 107,323 bits, PSNR = 30.96 dB). **e** Goudurix (Payload = 99463 bits, PSNR = 29.73 dB). **f** Manix (Payload = 109,795 bits, PSNR = 31.06 dB). **g** Phenix (Payload = 118,348 bits, PSNR = 28.04 dB). **h** Vix (Payload = 115,369 bits, PSNR = 31.42 dB)

method can avoid the blocking effects effectively, and Fig. 11 shows the superiority of this work in terms of Payload versus PSNR.

Conclusion

In summary, the proposed method has three advantages. First, all image blocks are classified as one of four separate types, and different data-embedding strategies are used for each type of block. Second, for blocks with AMBTC compression, the edge characteristics can be preserved, and for blocks with GB compression, the gradient characteristics can be preserved. By combining AMBTC compression and GB compression, undesirable block artifacts can be significantly alleviated. Third, our method has a larger payload than other methods, because it has greater capacity in gradient blocks than do other methods. As long as the number of gradient blocks is sufficient, the payload will be large. In the future, we plan to extend the idea of this work to different formats of images. For example, methods of [30,31] discuss about the possibility of embedding data in halftone images. In addition, currently, the proposed method is an irreversible hiding method; however, reversible hiding methods such as [32,33] have demonstrated their advantages of recovering the original image. We are trying to design a reversible hiding method which utilizes the gradient property.

Acknowledgements This work was supported in part by the Ministry of Science and Technology (MOST 108-2221-E-011-162-MY2 and 109-2628-E-011-002-MY2).

References

1. Yang Y, Xiao X, Cai X, Zhang W (2019) A secure and high visual-quality framework for medical images by contrast-enhancement reversible data hiding and homomorphic encryption. IEEE Access 7:96900–96911
2. Wang X, Bai L, Yang Q, Wang L, Jiang F (2019) A dual privacy-preservation scheme for cloud-based ehealth systems. J Inf Secur Appl 47:132–138
3. Qiu Y, Ying Q, Lin X, Zhang Y, Qian Z (2020) Reversible data hiding in encrypted images with dual data embedding. IEEE Access 8:23209–23220
4. Qian Z, Zhou H, Zhang X, Zhang W (2018) Separable reversible data hiding in encrypted JPEG bitstreams. IEEE Trans Depend Secure Comput 15(6):1055–1067
5. Yang Y, Xiao X, Cai X, Zhang W (2020) A secure and privacy-preserving technique based on contrast-enhancement reversible data hiding and plaintext encryption for medical images. IEEE Signal Process Lett 27:256–260
6. He J, Chen J, Luo W, Tang S, Huang J (2019) A novel high-capacity reversible data hiding scheme for encrypted JPEG bitstreams. IEEE Trans Circuits Syst Video Technol 29:3501–3515
7. Chen Y, Hung T, Hsieh S, Shiu C (2019) A new reversible data hiding in encrypted image based on multi-secret sharing and lightweight cryptographic algorithms. IEEE Trans Inf Forens Secur 14:3332–3343
8. Chen W, Chang C, Weng S, Ou B (2020) Multi-layer mini-sudoku based high-capacity data hiding method. IEEE Access 8:69256–69267
9. He J, Chen J, Tang S (2020) Reversible data hiding in JPEG images based on negative influence models. IEEE Trans Inf Forensics Secur 15:2121–2133
10. Xiao D, Li F, Wang M, Zheng H (2020) A novel high-capacity data hiding in encrypted images based on compressive sensing progressive recovery. IEEE Signal Process Lett 27:296–300
11. Yu Z, Lin C, Chang C (2020) ABMC-DH: an adaptive bit-plane data hiding method based on matrix coding. IEEE Access 8:27634–27648
12. Ding H, Li Z, Yang Y, You F, Liu F (2018) High quality data hiding in halftone image based on block conjugate. Chin J Electron 27:150–158
13. Li S, Zhang X (2019) Toward construction-based data hiding: from secrets to fingerprint images. IEEE Trans Image Process 28:1482–1497
14. Chen S, Chang C, Liao Q (2020) Fidelity preserved data hiding in encrypted images based on homomorphism and matrix embedding. IEEE Access 8:22345–22356
15. Horng J, Xu S, Chang C, Chang C (2020) An efficient data-hiding scheme based on multidimensional mini-SuDoKu. Sensors 20:2739
16. Liu Y, Yang C, Sun Q (2018) Enhance embedding capacity of generalized exploiting modification directions in data hiding. IEEE Access 6:5374–5378
17. Shukla A, Shukla A, Singh B, Kumar A (2018) A secure and high-capacity data-hiding method using compression, encryption and optimized pixel value differencing. IEEE Access 6:51130–51139
18. Lema M, Mitchell O (1984) Absolute moment block truncation coding and its application to color images. IEEE Trans Commun 32:1148–1157
19. Su G, Chang C, Lin C (2020) A high capacity reversible data hiding in encrypted AMBTC-compressed images. IEEE Access 8:26984–27000
20. Lin J, Lin C, Chang C (2019) Reversible steganographic scheme for AMBTC-compressed image based on (7,4) hamming code. Symmetry 11:1236
21. Zheng W, Chang C, Weng S (2020) A novel adjustable rdh method for ambtc-compressed codes using one-to-many map. IEEE Access 8:13105–13118
22. Chen C, Chang C, Lin C, Su G (2019) TSIA: a novel image authentication scheme for AMBTC-based compressed images using turtle shell based reference matrix. IEEE Access 7:149515–149526
23. Malik A, Sikka G, Verma H (2018) An AMBTC compression based data hiding scheme using pixel value adjusting strategy. Multidimension Syst Signal Process 29:1801–1818

24. Ou D, Sun W (2015) High payload image steganography with minimum distortion based on absolute moment block truncation coding. Multimedia Tools Appl 74:9117–9139
25. Malik A, Sikka G, Verma H (2019) A high capacity data hiding scheme using modified AMBTC compression technique. Int Arab J Inf Technol 16:148–155
26. Hong W (2018) Efficient data hiding based on block truncation coding using pixel pair matching technique. Symmetry 10:36
27. Kumar R, Kim D, Jung K (2019) Enhanced AMBTC based data hiding method using hamming distance and pixel value differencing. J Inf Secur Appl 47:94–103
28. Chang C, Chang X, Horng J (2019) Hybrid data hiding method for strict AMBTC format images with high-fidelity. Symmetry 11:1314
29. Osirix Database. https://www.osirix-viewer.com/support/knowledge-base/
30. Chen Y, Chi K, Hua K (2017) Design of image barcodes for future mobile advertising. EURASIP J Image Video Process 1:1–12
31. Chen Y, Chen W (2018) High-quality blind watermarking in halftones using random toggle approach. Multimedia Tools Appl 77:8019–8041
32. Lin Y, Hsia C, Chen B, Chen Y (2019) Visual IoT security: data hiding in AMBTC images using block-wise embedding strategy. Sensors 19:1974
33. Wang H, Lin H, Gao X, Cheng W, Chen Y (2019) Reversible AMBTC-based data hiding with security improvement by chaotic encryption. IEEE Access 7:38337–38347