Steganography Using Quotient Value Differencing and LSB Substitution for AMBTC Compressed Images

JI-HWEI HORNG, CHIN-CHEN CHANG, (Fellow, IEEE), AND GUAN-LONG LI

1Department of Electronic Engineering, National Quemoy University, Kinmen 89250, Taiwan
2Department of Information Engineering and Computer Science, Feng Chia University, Taichung 40724, Taiwan
3School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China

Corresponding author: Chin-Chen Chang (alan3c@gmail.com)

ABSTRACT In this paper, we apply quotient value differencing (QVD) and least significant bits (LSBs) substitution to hide secret data in an absolute moment block truncation coded (AMBTC) image. An AMBTC compressed image block consists of a high mean, low mean, and bitmap matrix. Based on the data structure of an AMBTC block, the proposed method includes three phases of embedding. In the first phase, QVD and LSB substitution are applied to embed secret digits into the low mean and high mean values. In the second phase, an additional bit is embedded by alternating the order of the low mean and high mean or not. Secret digits are embedded in the third phase by substituting the bitmap of smooth blocks. By leveraging the entire data structure of the AMBTC compressed code, the proposed method outperforms related works in embedding capacity.

INDEX TERMS Data hiding, image compression, AMBTC, QVD.

I. INTRODUCTION

With the increasing usage of Internet services, information security has attracted greater attention. Data hiding is a technology used to embed secret messages into some types of reliable media, such as videos, audio files, text files, and images, to conceal activity. These media are meaningful in themselves, and wiretapping cannot be aware of the embedding. Data hiding techniques can be classified into the following two categories: reversible and nonreversible data hiding. Reversible data hiding methods can recover the original cover media after the secret message is extracted. However, the embedding capacity of reversible data hiding methods is limited to a very low level compared with nonreversible methods.

Nonreversible data hiding can be further divided into the following three major types: LSB substitution [1]–[3], pixel value differencing (PVD) [4]–[6], and reference matrix (RM)-based [7]–[9] methods. The LSB substitution method embeds secret data by substituting one or more LSBs of the pixel values. These methods have a good embedding capacity. However, the image quality rapidly degrades when increasing the leveraged number of bits per pixel. Furthermore, the LSB substitution method is vulnerable to RS steganalysis [10]. The PVD method embeds data by evenly enlarging or reducing the difference value of a cover pixel pair, which improves the image quality at the expense of reducing the embedding capacity. In addition, the data can be detected by analyzing the pixel value difference histogram [11]. The RM-based method embeds data using a reference matrix. The information on constructing the RM should be securely shared between the encryption and decryption ends, which requires additional resources.

In 1989, Turner [2] proposed an LSB substitution method. This method has a good embedding capacity but suffers from image quality degradation with increased embedding. Wu and Tsai proposed a pixel-value differencing method [4] in 2003. The visual quality of a stego image is improved at the expense of reducing the embedding capacity to 0.2 bpp. In 2018, Jung mixed PVD and LSB substitution [12] to improve the embedding capacity. In his method, the image pixels are divided into a higher bit plane and a lower bit plane. Then, PVD is applied to embed the higher bit plane, while LSB substitution is applied to embed the lower bit plane. This
approach significantly improves the embedding capacity. However, the fall off boundary problem (FOBP) may occur during embedding. In 2018, Swain proposed an improved version [13] to solve the FOBP. Upon FOBP occurrence, LSB substitution is applied to the higher bit plane as well, instead of PVD. A marker bit is embedded in the lower bit plane to discriminate the different situations. Additionally, the PVD of the higher bit plane is renamed as quotient value differencing (QVD) in his work.

To transmit images efficiently, many image compression methods [14]–[17] have been proposed. In 1979, Delp and Mitchell proposed the block truncation coding method [16]. In this method, the original image is divided into blocks of size \( n \times n \). For each block, the mean value and standard deviation are calculated. A bit map is used to record the block, where the pixels are represented by ‘1’ when their value is greater than the mean and represented by ‘0’ otherwise. Two output levels are computed using the mean and standard deviation to substitute ‘1’ and ‘0’ in decompression. In 1984, Lema and Mitchell proposed an absolute moment block truncation coding method [17] to improve the quality and decompressed image.

To achieve data embedding and image compression at the same time, many AMBTC-based data hiding methods have been proposed [18]–[23]. In 2015, Ou et al. classified image blocks into smooth and complex blocks [20]. For smooth blocks, secret data are embedded directly into the bitmap, while embedding is performed through order exchange of two quantization levels for complex blocks. In 2019, Li et al. applied a modulo-2 operation in bitmap embedding of smooth blocks [22]. An additional 2 bits are embedded into the two quantization levels by applying the Hamming code (3, 2) technique. In the same year, Kumar et al. classified image blocks into smooth, less complex, and highly complex blocks [23]. For these three types of blocks, the bit substitution, Hamming distance calculation, and PVD are used to embed secret data, respectively.

In this paper, we focus on the extension of the payload for AMBTC-based data hiding. Since AMBTC images are highly compressed objects, to achieve this goal, we fully leverage the data structure of the image. AMBTC divides an image into nonoverlapping blocks, and each block is represented by two quantization levels and a bitmap. Three objectives of a block can be targeted, namely, the quantization levels, the order of the quantization levels, and the bitmap. The proposed data hiding scheme consists of three phases to deal with the three objectives.

By leveraging the full data structure of an AMBTC compressed block, our data hiding scheme can achieve a better embedding capacity than those described in related works. The remainder of this paper is organized as follows. In Section 2, we briefly review the AMBTC image format and the QVD-based data hiding proposed in [13]. In Section 3, the proposed scheme is introduced. Experimental results are given in Section 4. In Section 5, conclusions are drawn.

II. RELATED WORKS

In this section, we introduce two related works. The first is the AMBTC image compression method [17] proposed by Lema and Mitchell. The second is data embedding based on the QVD technique proposed by Swain [13].

A. ABSOLUTE MOMENT BLOCK TRUNCATION CODING (AMBTC)

AMBTC is a lossy compression technique for gray level images. The grayscale image is divided into nonoverlapping blocks before compression. Then, each block is represented by two quantization levels and a bitmap. The details of the encoding process are as follows.

The input image of size \( W \times H \) is divided into nonoverlapping blocks of size \( n \times n \). To compress an image block, the mean value \( \mu \) is calculated first by (1)

\[
\mu = \frac{1}{n \times n} \sum_{i=1}^{n \times n} p_i, \quad (1)
\]

where \( p_i \) is the gray level value of the pixel in the block. Then, each pixel value is compared with the mean, assuming the number of pixels greater than or equal to the mean is \( m \). The two quantization levels are calculated by (2) and (3).

\[
\mu H = \frac{1}{m} \sum_{i \geq p_i} p_i, \quad (2)
\]

\[
\mu L = \frac{1}{n \times n - m} \sum_{i < p_i} p_i. \quad (3)
\]

A bitmap \( BM \) of size \( n \times n \) is constructed, where the bit value is ‘0’ when its corresponding pixel value is less than the mean, while the bit value is ‘1’ otherwise. Finally, each image block is recorded by \( (L, H, BM) \) in the compressed code. The decoding process is to construct an image block of size \( n \times n \), where the pixel value is \( L \) when the corresponding value in the bitmap is ‘0’ and is \( H \) when the corresponding value is ‘1’.

B. DATA HIDING SCHEME BASED ON QVD AND LSB SUBSTITUTION

In the data hiding scheme proposed by Swain [13], the image is partitioned into nonoverlapping blocks of \( 3 \times 3 \) pixels. Each pixel block is sheared into a quotient block (i.e., higher bit-plane block) and an LSB block (i.e., lower bit-plane block) by (4) and (5).

\[
Q_i = P_i \mod 4, \quad i = 1 \text{ to } 8 \quad \text{and} \quad Q_c = P_c \mod 4, \quad (4)
\]

\[
R_i = P_i \div 4, \quad i = 1 \text{ to } 8 \quad \text{and} \quad R_c = P_c \div 4, \quad (5)
\]

where ‘\( \mod \)’ and ‘\( \div \)’ are the divide and modulo operations, respectively. The resulting blocks are shown in Fig. 1.

The lower bit-plane is embedded by substitution directly in the bitmap, while embedding is performed through order exchange of two quantization levels for complex blocks.

\[
d_i = Q_i - Q_c, \quad \text{for} \quad i = 1, 2, \ldots, 8. \quad (6)
\]
The embedding capacity for a quotient value $Q_i$ is determined by comparing its absolute difference $|d_i|$ with the range table given in Table 1. Supposing $|d_i|$ falls in the range $r_j = [r_j^L, r_j^H]$, $n_j$ data bits are collected and converted into decimal value $b_i$. Then, the new difference value $d'_i$ and the new quotient pairs $(Q_{ci}, Q'_i)$ are calculated by (7) and (8).

$$d'_i = \begin{cases} r_j^L + b_i, & \text{if } d_i \geq 0, \\ -r_j^L - b_i, & \text{if } d_i < 0. \end{cases}$$ (7)

$$(Q_{ci}, Q'_i) = \begin{cases} Q_c - \frac{(d'_i - d_i)}{2}, & \text{if } d_i \text{ is even,} \\ Q_c + \frac{(d'_i - d_i)}{2}, & \text{if } d_i \text{ is odd.} \end{cases}$$ (8)

However, we cannot assign eight different values to the new central quotient $Q'_c$. The compromise is to average them by (9).

$$Q'_c = \left[\frac{1}{8} \sum_{i=1}^{8} Q'_{ci}\right].$$ (9)

Then, $Q'_i$ is adjusted again to fit the required $d'_i$ by (10).

$$Q'_i = Q'_c + (Q'_c - Q'_i), \text{ for } i = 1, 2, \ldots, 8. \quad (10)$$

If there is no FOBP, i.e., $Q'_i \in [0, 63]$, the marker bit is set to ‘1’. The quotient block and the LSB block are merged to obtain the stego block. Otherwise, 4-bit LSB substitution is applied to the covered pixels directly, and the marker bit is set to ‘0’.

The extraction process is in the reverse order of embedding. The marker bit is extracted first to determine the type of embedding. Then, the secret data are extracted according to the embedding type.

III. THE PROPOSED SCHEME

In this section, we propose a new data hiding scheme for AMBTC images. The proposed scheme consists of three phases to fully leverage the AMBTC data structure. The data hiding techniques of QVD, LSB substitution, quantization level swapping, and bitmap replacement are hybrid to achieve a high embedding capacity. The data embedding and extraction flowchart is shown in Fig. 2. The details are presented in the following subsections.

A. EMBEDDING PROCEDURE

In the first phase of embedding, the two quantization levels of the AMBTC code are used to hide the secret data. The QVD and LSB substitution approach [13] proposed by Swain is applied. The quantization levels $(L, H)$ are sheared into quotients and LSBs by (11) and (12).

$$Q_L = L \div 4 \quad \text{and } Q_H = H \div 4, \quad (11)$$
$$R_L = L \mod 4 \quad \text{and } R_H = H \mod 4. \quad (12)$$

Then, the quotient value difference is calculated by (13).

$$d = Q_H - Q_L. \quad (13)$$

The embedding capacity for the quotient pair is determined by comparing its absolute difference $|d|$ with the modified range table given in Table 2. Suppose $|d|$ falls in the range $r_j = [r_j^L, r_j^H]$, $n_j$ data bits are collected and converted to decimal value $b$. Then, the new difference value $d'$ and the new quotient pairs $(Q'_L, Q'_H)$'s are calculated by (14) and (15).

$$d' = r_j^L + b. \quad (14)$$

$$\begin{align*}
Q'_L &= L \div 4, \\
Q'_H &= H \div 4, \\
R'_L &= L \mod 4, \\
R'_H &= H \mod 4.
\end{align*} \quad (15)$$
Since a quotient is the six most significant bits of a quantization level, direct bit substitution will result in severe degradation of the stego image quality. To increase the payload with a slight quotient modification, the QVD is a good option. The quotient value difference is divided into five ranges. The width of each range is decided by the number of bits to be embedded, i.e., $r_H^j - r_L^j + 1 = 2^{n_j}$.

A decimal value $b$ of $n_j$ data bits is embedded by adjusting the quotient value difference to $d' = r_L^j + b$. While $d'$ is achieved by evenly expanding or shrinking the two quantization levels, on average, the required modification of the quantization levels is significantly lower than direct bit substitution. In addition, the human visual system has a larger tolerance to degradation of high contrast blocks. Therefore, the adjustment range is designed to increase with the quotient value difference.

For LSBs, i.e., $(R_L, R_H)$, LSB substitution is applied and produces $(R'_L, R'_H)$. For rare cases of FOBP, the quotients are shifted back to fit the boundary by (16).

Thus, the marker bit used in [13] is not required here. The final process in the first phase is to merge the quotients and LSBs back to the modified quantization levels by (17).

In the second phase of embedding, a single secret bit is embedded. If the secret bit is ‘1’, the order of the quantization levels is swapped and the bitmap is flipped, i.e., $(L', H', BM) \rightarrow (H', L', BM)$; otherwise, the compressed code remains unchanged. For the special case of $Q'_H = Q'_L$, this block is nonembeddable in the second phase.

In the third phase, the modified difference $d_p = H' - L'$ is compared with the predetermined threshold $d_{th}$. If $d_p < d_{th}$, then the bitmap is replaced by secret bits; otherwise, the bitmap is nonembeddable. A compressed block of very close quantization levels that cannot be discriminated by a human visual system is defined as a smooth block and can be totally replaced by secret data. On the contrary, a block with a difference greater than or equal to the threshold is defined as a complex block and is nonembeddable. The complete data embedding algorithm is summarized as follows.

**B. EXTRACTION PROCEDURE**

The data extraction procedure is in the reverse order of embedding. At the receiver end, the stego AMBTC code and the threshold $d_{th}$, which is shared with the transmitter end, are used to extract the secret data. For each stego AMBTC block, the quantization levels are compared first. If the difference is
Data Embedding Algorithm

Input: cover image \(I\), secret data \(S\), threshold \(d_{th}\)
Output: stego image in AMBTC format \(I'\)

Step 1: compress the cover image \(I\) into AMBTC format.
Step 2 (phase 1): shear the quantization levels \((L, H)\) into \((Q_L, Q_H)\) and \((R_L, R_H)\).
Step 3: apply Eqs. (13)-(16) to \((Q_L, Q_H)\) and obtain \((Q'_L, Q'_H)\).
Step 4: apply LSB substitution to \((R_L, R_H)\) and obtain \((R'_L, R'_H)\).
Step 5: merge \((Q'_L, Q'_H)\) and \((R'_L, R'_H)\) back to \((L', H')\).
Step 6: (phase 2): if \(Q'_H > Q'_L\), embed ‘1’ by \((L', H', BM)\) → \((H', L', BM)\); keep unchanged for ‘0’; leave non-embedded for \(Q'_H = Q'_L\).
Step 7 (phase 3): If \(|d_p| = |H' - L'| < d_{th}\), replace BM by secret bits.
Step 8: write modified block to \(I'\) and repeat Steps 2-7, until all blocks are embedded.

C. EXAMPLES OF EMBEDDING AND EXTRACTION

In this subsection, we give some examples to demonstrate the proposed data embedding and extraction procedures. In our examples, the threshold is set to 8, i.e., \(d_{th} = 8\). The first example is a demonstration of the data embedding procedure and is illustrated in Fig. 3. The cover block in 3(a) is compressed to the AMBTC format (142, 146, BM), as shown in 3(b). Then, the quantization levels (142, 146) are sheared into \((Q_L, Q_H) = (35, 36)\) and \((R_L, R_H) = (2, 2)\). The quotient difference calculated by \(d = 36 - 35 = 1\). According to Table 2, the value fits the range \(r_1 = [0, 3]\) and 2 bits of secret (10) are embedded by calculating \(d' = (10)_2 = 2\) and

\[
(Q'_L, Q'_H) = (35 - \left\lfloor \frac{2 - 1}{2} \right\rfloor , 36 + \left\lceil \frac{2 - 1}{2} \right\rceil) = (35, 37).
\]

LSB substitution is applied to \((R_L, R_H)\) by \((R'_L, R'_H) = ((11)_2, (01)_2) = (3, 1)\). Then, \((L', H') = (Q'_L \times 4 + R'_L, Q'_H \times 4 + R'_H) = (35 \times 4 + 3, 37 \times 4 + 1) = (143, 149)\) is merged back, as shown in 3(c). In the second phase of embedding, since \(Q'_H > Q'_L\), a secret bit (0) is embedded by no change, as shown in 3(d). In the last embedding phase, since

![Data Embedding Algorithm](image-url)
\[ d_p = H' - L' = 149 - 143 = 6 < 8 = d_{th}, \] the bitmap is replaced by secret bits, as shown in 3(e).

The second example is illustrated in Fig. 4. The cover block in 4(a) is compressed to the AMBTC format (43, 82, BM), as shown in 4(b). Then, the quantization levels (43, 82) are sheared into \( (Q_L, Q_H) = (10, 20) \) and \( (R_L, R_H) = (3, 2) \). The quotient difference is calculated by \( d = 20 - 10 = 10 \). According to Table 2, the value fits the range \( r_3 = [8, 15] \), 3 bits of secret (110)\(_2\) is embedded by calculating \( d' = 8 + (110) = 14 \), and

\[ (Q_L', Q_H') = (10 - \left[ \frac{14 - 10}{2} \right], 20 + \left[ \frac{14 - 10}{2} \right]) = (8, 22). \]

LSB substitution is applied to \( (R_L, R_H) \) by \( (R_L', R_H') = ((10)_2, (01)_2) = (2, 1) \). Then, \( (L', H') = (Q_L' \times 4 + R_L', Q_H' \times 4 + R_H') = (8 \times 4 + 2, 22 \times 4 + 1) = (34, 89) \) is merged back, as shown in 4(c). In the second phase of embedding, \( Q_H' > Q_L' \) is checked, and a secret bit (1)\(_2\) is embedded by swapping (34, 89) to (89, 34) and flipping the bitmap as shown in 4(d). In the last embedding phase, since \( d_p = H' - L' = 89 - 34 = 55 > 8 = d_{th}, \) it is a complex block, and the bitmap is nonembeddable.

The third example is a demonstration of the data extraction procedure for smooth blocks. As shown in Fig. 5, the difference of the modified quantization levels is calculated first, i.e., \( d_p = 143 - 149 = -6 \). Since \( |d_p| < d_{th}, \) the block is a smooth block, and the bitmap is extracted as s3. Then, the quantization levels (143, 149) are sheared into quotients (35, 37) and LSBs (3, 1). Since \( Q_A < Q_B, \) a secret bit \( s_2 = 0 \) is extracted. Next, \( s_11 = (Q_B - Q_A - r_{ij}^L) \)
FIGURE 6. Example of data extraction for complex blocks.

$\left(37 - 35 - 0\right)_2 = \left(10\right)_2, s_{12} = (R_A)_2 = (3)_2 = (11)_2, s_{13} = (R_B)_2 = (1)_2 = (01)_2$ are calculated, and thus, $s_1 = (1011010)$. The secrets are concatenated into $s = (s_1 || s_2 || s_3)$, as shown in Fig. 5. The last example given in Fig. 6 is an illustration of data extraction for complex blocks. The difference of the modified quantization levels and their absolute value are calculated and checked first, i.e., $d_p = 89 - 34 = 55$ and $|d_p| > d_{th} = 8$. Thus, $s_3$ is empty and skipped. Then, the quotients and LSBs are computed to obtain $(22, 8)$ and $(1, 2)$, respectively. Since $Q_A > Q_B$, $s_2 = 1$ is extracted, two quantization levels are swapped and the bitmap is flipped.

Next, $s_{11} = (Q_A - Q_B - r^j_L)_2 = (22 - 8 - 8)_2 = (110)_2, s_{12} = (R_B)_2 = (2)_2 = (10)_2, s_{13} = (R_A)_2 = (1)_2 = (01)_2$ are calculated, and thus, $s_1 = (110100101)$. The overall secret data embedded in this block is therefore $s = (s_1 || s_2)$.

IV. EXPERIMENTAL RESULTS

In our experiments, MATLAB R2017a is used to implement the proposed data hiding scheme. Six standard gray level images (see Fig. 7) of size $512 \times 512$ are used, including Air-
plane, Baboon, Boats, Lena, Peppers, and Zelda. Secret data are produced by a random number generator. The hardware and operating system are an Intel® Core™ i7-3770 CPU @ 3.40 GHz, 8 GB RAM PC and Windows 10 Professional, respectively.

Since the proposed data hiding scheme is based on the AMBTC compressed images, we have to evaluate the image quality of the compressed images first. The peak signal to noise ratio (PSNR), defined by (18) and (19), is used to measure the visual quality after compression.

\[
\text{PSNR} = 10 \log_{10} \frac{255^2}{\text{MSE}},
\]

\[
\text{MSE} = \frac{1}{N \times N} \sum_{i=1}^{N} \sum_{j=1}^{N} (I_{i,j} - I'_{i,j})^2,
\]

where MSE is the mean square error between the cover image \( I \) and the AMBTC compressed image \( I' \); the image size is \( N \times N \), i.e., \( N = 512 \) in our experiments.

The test images are divided into nonoverlapping blocks of size \( 4 \times 4 \) and compressed in the AMBTC format. For each \( 4 \times 4 \) block, the compressed code uses two quantization levels of one byte each and a bitmap of sixteen bits to represent the sixteen bytes of the pixel values. Therefore, the compression ratio is 1:4. The images recovered from the AMBTC compressed format are shown in Fig. 8, which serve as the input of our embedding scheme. The PSNR values with respect to the original standard images are also given below each image. Typically, the difference cannot be discriminated by the human visual system and provide a value greater than 30 dB.

The data hiding capacity and stego image quality of the proposed scheme depend on the threshold value \( d_{th} \) of the block classification. Increasing the threshold value will obtain more smooth blocks and thus a higher embedding capacity. At the same time, the visual quality of the stego image will degrade gradually. The results of applying the proposed scheme with different threshold values to the six test images are listed in Table 3.

Based on the initial PSNR of each cover image, the visual quality of a stego image degrades with the increasing hiding capacity controlled by the threshold value. At a threshold value of 16, our data hiding scheme can embed approximately 1.2 bits per pixel (bpp) with a PSNR of approximately 30 dB. The image ‘Baboon’ is a special case initiated from an input image of PSNR 27.77 dB.

FIGURE 8. The cover images reconstructed from AMBTC compressed test images.
TABLE 3. Experimental results of the proposed scheme with different threshold values.

| Image   | Metrics | $d_{th} = 0$ | $d_{th} = 4$ | $d_{th} = 8$ | $d_{th} = 16$ | $d_{th} = 32$ |
|---------|---------|--------------|--------------|--------------|--------------|--------------|
| Airplane| PSNR    | 31.65        | 30.78        | 30.56        | 29.91        | 28.4         |
|         | capacity| 114666       | 193883       | 242106       | 313825       | 341283       |
|         | bpp     | 0.43         | 0.73         | 0.92         | 1.19         | 1.30         |
| Baboon  | PSNR    | 26.39        | 26.33        | 26.29        | 25.97        | 24.32        |
|         | capacity| 126600       | 153308       | 172835       | 210959       | 276510       |
|         | bpp     | 0.48         | 0.58         | 0.65         | 0.80         | 1.05         |
| Boat    | PSNR    | 30.59        | 29.89        | 29.69        | 29.05        | 27.30        |
|         | capacity| 115448       | 184708       | 227939       | 294787       | 335286       |
|         | bpp     | 0.44         | 0.70         | 0.86         | 1.12         | 1.27         |
| Lena    | PSNR    | 31.42        | 31.36        | 31.05        | 30.16        | 28.93        |
|         | capacity| 124877       | 192923       | 241605       | 315188       | 348178       |
|         | bpp     | 0.47         | 0.73         | 0.92         | 1.20         | 1.32         |
| Pepper  | PSNR    | 31.72        | 31.51        | 31.23        | 30.29        | 28.74        |
|         | capacity| 125657       | 198471       | 249283       | 326352       | 352396       |
|         | bpp     | 0.47         | 0.75         | 0.95         | 1.24         | 1.34         |
| Zelda   | PSNR    | 33.42        | 33.38        | 32.86        | 31.49        | 29.56        |
|         | capacity| 125447       | 199325       | 255131       | 336462       | 362920       |
|         | bpp     | 0.47         | 0.76         | 0.97         | 1.28         | 1.38         |
| Average | PSNR    | 30.86        | 30.54        | 30.28        | 29.47        | 27.87        |
|         | capacity| 122115       | 187103       | 231483       | 299595       | 336095       |
|         | bpp     | 0.46         | 0.71         | 0.88         | 1.14         | 1.28         |

FIGURE 9. Performance comparisons of test images with the proposed scheme.

Fig. 9 shows the relationship between the embedding capacity and stego image quality. As the embedding capacity increases with the threshold of block classification, the image quality degrades gradually. In addition, the image feature can be identified by the level of PSNR. The PSNR level indicates the smoothness of the applied image. The curve for the smoothest image ‘Zelda’ is at the top, while that for the most complex image ‘Baboon’ is at the bottom.

Table 4 compares the experimental data of the proposed scheme with related works, including those by Ou and Sun [20], Li et al. [22], and Kumar et al. [23]. Since different data hiding schemes are controlled by different parameters, the embedding capacities are not exactly the same. However, the highlighted entries indicate that the proposed scheme outperforms the methods proposed in related works at high payloads.

A graphical representation of Table 4 is shown in Fig. 10, which indicates that our scheme has no advantage at a low payload. However, as the payload increases, our scheme gradually outperforms others and reaches the maximum payload among all the schemes.
### TABLE 4. Performance comparisons of the proposed scheme with related works.

|          | Proposed | Kumar et al. [20] | Li et al. [22] | Ou et al. [23] |
|----------|----------|-------------------|----------------|----------------|
|          | EC       | PSNR              | EC             | PSNR           | EC              | PSNR           |
| **Airplane** |          |                   |                |                |                 |                |
| 114666   | 31.65    | 212684            | 30.37          | 169726         | 31.61           | 178099         | 31.91          |
| 193883   | 30.78    | 230434            | 29.78          | 220139         | 31.09           | 209794         | 31.34          |
| 242106   | 30.56    | 239510            | 29.11          | 244390         | 30.54           | 223039         | 30.75          |
| 313825   | 29.91    | 245927            | 28.41          | 260268         | 29.95           | 233329         | 29.97          |
| 341283   | 28.40    | 250541            | 27.92          | 270109         | 29.44           | 240709         | 29.20          |
| **Baboon** |          |                   |                |                |                 |                |
| 126600   | 26.39    | 140130            | 27.23          | 21280          | 26.97           | 50374          | 26.92          |
| 153308   | 26.33    | 178272            | 26.24          | 62142          | 26.82           | 106534         | 26.58          |
| 172835   | 26.29    | 208525            | 25.08          | 102758         | 26.47           | 141919         | 26.02          |
| 210959   | 25.97    | 232516            | 24.55          | 131196         | 26.07           | 173974         | 25.17          |
| 276510   | 24.32    | 247766            | 23.85          | 154624         | 25.57           | 204319         | 24.11          |
| **Boat** |          |                   |                |                |                 |                |
| 115448   | 30.59    | 215485            | 31.43          | 132422         | 32.79           | 172579         | 32.67          |
| 184708   | 29.89    | 237743            | 30.61          | 215463         | 31.78           | 216529         | 31.78          |
| 227939   | 29.69    | 248156            | 29.89          | 251063         | 30.88           | 234004         | 30.91          |
| 294787   | 29.05    | 254310            | 29.35          | 270236         | 30.13           | 245569         | 30.02          |
| 335286   | 27.30    | 257754            | 29.05          | 282952         | 29.44           | 252154         | 29.30          |
| **Lena** |          |                   |                |                |                 |                |
| 124877   | 31.42    | 182577            | 29.20          | 138369         | 30.73           | 148609         | 31.32          |
| 192923   | 31.36    | 224100            | 28.58          | 184780         | 30.71           | 191914         | 30.62          |
| 241605   | 31.05    | 238456            | 27.89          | 207205         | 29.81           | 217264         | 29.69          |
| 315188   | 30.16    | 246979            | 27.26          | 232767         | 29.02           | 231544         | 28.83          |
| 348178   | 28.93    | 252678            | 26.81          | 250673         | 28.29           | 241834         | 28.00          |
| **Pepper** |         |                   |                |                |                 |                |
| 125657   | 31.72    | 234737            | 34.19          | 101124         | 33.89           | 191102         | 33.70          |
| 198471   | 31.51    | 253254            | 33.54          | 220179         | 33.47           | 228688         | 32.60          |
| 249283   | 31.23    | 259230            | 33.10          | 263345         | 33.21           | 242132         | 32.10          |
| 326352   | 30.29    | 261101            | 32.88          | 281015         | 32.89           | 252138         | 31.30          |
| 352396   | 28.74    | 261918            | 32.85          | 292134         | 32.02           | 259458         | 30.70          |
| **Zelda** |          |                   |                |                |                 |                |
| 125447   | 33.42    | 221134            | 31.36          | 94902          | 33.05           | 183529         | 33.36          |
| 199325   | 33.38    | 239832            | 30.83          | 219291         | 31.60           | 224989         | 32.42          |
| 255131   | 32.86    | 247698            | 30.24          | 260514         | 30.61           | 238969         | 31.64          |
| 336462   | 31.49    | 252296            | 29.67          | 278018         | 29.98           | 247219         | 30.88          |
| 362920   | 29.56    | 255360            | 29.33          | 288044         | 29.44           | 251854         | 30.29          |
V. CONCLUSIONS

In this paper, we propose a data hiding scheme for AMBTC compressed images. The QVD and LSB substitution techniques are applied to embed data in the quantization levels. For each pair of quantization levels, the proposed scheme can hide 6 to 9 bits of secret data. One additional bit is included for order arrangement of the quantization levels. For smooth blocks, the bitmap is totally replaced with the secret bits. By leveraging three phases of embedding, the proposed scheme achieves the best embedding capacity in comparison with existing AMBTC-based methods using the same PSNR level.
According to a series of related works, the performance of the three different strategies can be concluded as follows. The quantization level swapping with bitmap flipping will not degrade the PSNR; however, its payload is limited. Bitmap replacement and quantization level modification can achieve a higher payload, but the PSNR degrades with increasing payload.

As experimentally shown, modifying the quantization levels while preserving the bitmap of the low contrast block has no advantage. Therefore, the latter two strategies achieved better results. To design a data hiding scheme applicable to a broad range of payloads, swapping the quantization levels may be applied first. Then, the parameters of the latter two strategies are adjusted gradually to achieve the required payload. In this way, different payloads can be achieved with the least degradation of stego image quality.

REFERENCES

[1] J. Mielikainen, “LSB matching revisited,” IEEE Signal Process. Lett., vol. 13, no. 5, pp. 285–287, May 2006.
[2] L. P. Turner. (Jan. 2019). Digital Data Security System. [Online]. Available: http://en.wikipedia.org/wiki/Mathematics_of_Sudoku
[3] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber, “Lossless generalized-LSB data embedding,” IEEE Trans. Image Process., vol. 14, no. 2, pp. 253–266, Feb. 2005.
[4] D.-C. Wu and W.-H. Tsai, “A steganographic method for images by pixel-value differencing,” Pattern Recognit. Lett., vol. 24, nos. 9–10, pp. 1613–1626, Jun. 2003.
[5] C.-M. Wang, N.-I. Wu, C.-S. Tsai, and M.-S. Hwang, “A high quality steganographic method with pixel-value differencing and modulus function,” J. Syst. Softw., vol. 81, no. 1, pp. 150–158, Jan. 2008.
[6] W. Luo, F. Huang, and J. Huang, “A more secure steganography based on adaptive pixel-value differencing scheme,” Multimedia Tools Appl., vol. 52, nos. 2–3, pp. 407–430, Apr. 2011.
[7] C.-C. Chang, Y.-C. Chou, and T. D. Kieu, “An information hiding scheme using Sudoku,” in Proc. 3rd Int. Conf. Innov. Comput. Inf. Control, Jun. 2008, pp. 17–22.
[8] C. C. Chang, Y. Liu, and T. S. Nguyen, “A novel turtle shell based scheme for data hiding,” in Proc. 10th Int. Conf. Intell. Hiding Multimedia Signal Process., Kitakyushu, Japan, Aug. 2014, pp. 89–93.
[9] Y. Liu, C.-C. Chang, and T.-S. Nguyen, “High capacity turtle shell-based data hiding,” IET Image Process., vol. 10, no. 2, pp. 130–137, Feb. 2016.
[10] J. Fridrich and M. Goljan, “Practical steganalysis of digital images: State of the art,” Proc. SPIE, vol. 4675, pp. 1–13, Apr. 2002.
[11] X. Zhang and S. Wang, “Vulnerability of pixel-value differencing steganography to histogram analysis and modification for enhanced security,” Pattern Recognit. Lett., vol. 25, no. 3, pp. 331–339, Feb. 2004.
[12] K.-H. Jung, “Data hiding scheme improving embedding capacity using mixed PVD and LSB on bit plane,” J. Real-Time Image Process., vol. 14, no. 1, pp. 127–136, Jan. 2018.
[13] G. Swain, “Very high capacity image steganography technique using quotient value differencing and LSB substitution,” Arabian J. Sci. Eng., vol. 44, no. 4, pp. 2905–3004, Apr. 2019.
[14] J. Jav and A. Lempel, “A universal algorithm for sequential data compression,” IEEE Trans. Inf. Theory, vol. IT-23, no. 3, pp. 337–343, May 1977.
[15] D. Taubman, “High performance scalable image compression with EBCOT,” IEEE Trans. Image Process., vol. 9, no. 7, pp. 1158–1170, Jul. 2000.
[16] E. Delp and O. Mitchell, “Image compression using block truncation coding,” IEEE Trans. Commun., vol. COM-27, no. 9, pp. 1335–1342, Sep. 1979.
[17] M. Lema and O. Mitchell, “Absolute moment block truncation coding and its application to color images,” IEEE Trans. Commun., vol. COM-32, no. 10, pp. 1148–1157, Oct. 1984.
[18] J. S. Pan, W. Li, and C. C. Lin, “Novel reversible data hiding scheme for AMBTC-compressed images by reference matrix,” IEEE Trans. Image Process., vol. 23, no. 3, Mar. 2014, pp. 969–978.
[19] C.-C. Lin, X.-L. Liu, W.-L. Tai, and S.-M. Yuan, “A novel reversible data hiding scheme based on AMBTC compression technique,” Multimedia Tools Appl., vol. 74, no. 11, pp. 3823–3842, Jun. 2015.
[20] D. Ou and W. Sun, “High payload image steganography with minimum distortion based on absolute moment block truncation coding,” Multimedia Tools Appl., vol. 74, no. 21, pp. 9117–9139, Nov. 2015.
[21] Y.-H. Huang, C.-C. Chang, and Y.-H. Chen, “Hybrid secret hiding schemes based on absolute moment block truncation coding,” Multimedia Tools Appl., vol. 76, no. 5, pp. 6159–6174, Mar. 2017.
[22] L. Li, M. He, S. Zhang, T. Luo, and C. C. Chang, “AMBTC based high payload data hiding with modulo-2 operation and Hamming code,” Math. Biosci. Eng., vol. 16, no. 6, pp. 7934–7949, Aug. 2019.
[23] R. Kumar, D.-S. Kim, and K.-H. Jung, “Enhanced AMBTC based data hiding method using Hamming distance and pixel value differencing,” J. Inf. Secur. Appl., vol. 47, pp. 94–103, Aug. 2019.

JI-HWEI HORNG received the B.S. degree from the Department of Electronic Engineering, Tamkang University, Taipei, Taiwan, in 1990, and the M.S. and Ph.D. degrees from the Department of Electrical Engineering, National Taiwan University, Taipei, in 1992 and 1996, respectively. He was a Professor and the Chairman of the Department of Electronic Engineering, from 2006 to 2009, and the Dean of the College of Science and Engineering, from 2011 to 2014, with the National Quemoy University, Kinmen, Taiwan. He is currently the Vice President of Academic Affairs with NQU. His research interests include image processing, pattern recognition, information security, and artificial intelligence.

CHIN-CHEH CHANG (Fellow, IEEE) received the B.S. degree in applied mathematics, the M.S. degree in computer and decision sciences from National Tsing Hua University, and the Ph.D. degree in computer engineering from National Chiao Tung University. He was with National Chung Cheng University, from 1989 to 2005. He is currently the Chair Professor of the Department of Information Engineering and Computer Science, Feng Chia University. Prior to joining Feng Chia University, he was an Associate Professor with Chiao Tung University, a Professor with National Chung Hsing University, and a Chair Professor with National Chung Cheng University. He has also been a Visiting Researcher with Tokyo University, Japan, and a Visiting Scientist with Kyoto University, Japan. During his service in Chung Cheng, he has served as the Chairman of the Institute of Computer Science and Information Engineering, the Dean of the College of Engineering, a Provost, and then the Acting President of Chung Cheng University, and the Director of Advisory Office in Ministry of Education, Taiwan. His current research interests include database design, computer cryptography, image compression, and data structures. He has won many research awards and has honorary positions in prestigious organizations both nationally and internationally. Since his early years of career development, he has consecutively won awards, including the Outstanding Talent in Information Sciences of the R.O.C., the Acer Dragon Award of the Ten Most Outstanding Talents, the Outstanding Scholar Award of the R.O.C., the Outstanding Engineering Professor Award of the R.O.C., the Distinguished Research Awards of National Science Council of the R.O.C., and the Top Fifteen Scholars in Systems and Software Engineering of the Journal of Systems and Software. On numerous occasions, he has been invited to serve as a Visiting Professor, the Chair Professor, an Honorary Professor, the Honorary Director, the Honorary Chairman, a Distinguished Alumnus, a Distinguished Researcher, and a Research Fellow by various universities and research institutes.

GUAN-LONG LI received the B.S. degree in computer science from Feng Chia University, Taichung, Taiwan, in 2018, where he is currently pursuing the M.S. degree. His current research interests include cryptography, information security, watermarking, and information hiding.

* * *