# ADAPTIVE REVERSIBLE DATA HIDING METHOD USING BITMAP CLASSIFICATION STRATEGY FOR AMBTC-COMPRESSED IMAGES

*Duc Tuan Nguyen*[1,*]

**Abstract**

Reversible data hiding (RDH) has attracted significant attention from researchers. Absolute Moment Block Truncation Coding (AMBTC) compressed image is a popular cover medium in RDH schemes due to its simplicity and high compression ratio. Nevertheless, most AMBTC-based RDH schemes produce stego images that cannot be decoded by standard AMBTC decoders because they use reconstructed pixels to embed data. Although some AMBTC-based RDH schemes generate stego images compatible with AMBTC decoders, their embedding capacity (EC) is limited. To address this issue, in this paper, a high-capacity AMBTC-based RDH scheme compatible with standard decoders is proposed. In this scheme, an effective bitmap classification technique is developed and employed to ensure reversibility and enhance the embedding payload. Furthermore, the proposed scheme provides the ability to adaptively balance the trade-off between hiding the payload and the perceptual quality of stego images by utilizing two predefined thresholds. Experimental results demonstrate that the proposed scheme outperforms related approaches in both embedding payload and visual quality.

**Index terms**

Reversible data hiding; secure key; adaptive; location map; bitmap classification; AMBTC compressed images.

## 1. Introduction

Nowadays, in the digital era, the demand for transferring valuable information via the Internet has significantly increased. This information is vulnerable to security breaches if protection techniques are not applied. Data hiding acts as an additional layer of security for valuable data, complementing encryption through cryptographic algorithms. Digital images are the most popular cover medium used in data hiding methods because of their redundancy, which allows message bits to be embedded with minimal distortion. Data hiding techniques are divided into two categories: reversible and non-reversible

[1]Vietnam National University, Hanoi, School of Interdisciplinary Sciences and Arts.
[*]Corresponding author, email: tuannguyenduc@vnu.edu.vn

DOI: 10.56651/lqdtu.jst.v14.n01.1045.ict

data hiding. RDH methods enable the restoration of the stego image to its original form after the secret message has been extracted. To date, a variety of RDH approaches [1]–[5] have been developed. Both uncompressed and compressed image formats can be utilized in RDH techniques, but embedding data in compressed images is highly desirable. The AMBTC compressed image format, in particular, has increasingly attracted the attention of researchers due to the simplicity of its compression algorithm and the compactness of the resulting compressed data. AMBTC-based RDH schemes can be classified into two types: Type-I and Type-II.

Type-I AMBTC-based RDH methods [6]–[11] employ the AMBTC compressed code to embed message bits without altering the structure of the code. As a result, stego-AMBTC images generated by Type-I AMBTC-based RDH schemes can be decoded by traditional AMBTC decoders. Yin *et al.* [11] proposed an RDH in encrypted AMBTC images, which utilizes the higher mean $(H)$ and lower mean $(L)$ of the AMBTC compressed code to calculate a prediction error. The message bits are then embedded in the quantization levels $L$ based on the histogram of the estimated prediction error. The disadvantage of this method is its low EC since only the low mean values $L$ are used to carry the secret data. Furthermore, if the size of a compressed block is $(w \times h)$, the available EC is limited to $(\frac{imgSize}{w \times h})$, where $w$ and $h$ represent the width and height of the compressed block, respectively.

To increase EC, Chang *et al.* [10] introduced a novel AMBTC-based RDH scheme that employs both quantization level values, $H$ and $L$, to embed message bits. Although this technique enhances the EC of this approach, the structure of the compressed code (DS) is altered, as auxiliary bits are added to the elements $H$ and $L$, expanding their length from the standard 8 bits to 12 bits. As a result, the file size of the compressed AMBTC images is also increased. In 2020, Hui *et al.* [9] introduced a RDH approach using AMBTC-compressed images. In this scheme, AMBTC trios (DSs) are classified into two categories based on a given threshold: smooth trios and complex trios. Regarding smooth trios, a modified low quantization level is created by concatenating Huffman codes with secret bits to hide payload bits, whereas a reversible contrast mapping (RCM) is applied to complex trios for the same purpose. Consequently, the data hiding payload of this approach is enhanced, but a location map $A_1$ which has a size equal to the number of AMBTC trios, is yielded during the data hiding process. $A_1$ stores the original least significant bits (LSBs) of high quantization level values $H$, as this bit changed to indicate whether the considered trio is smooth or complex. In addition, another auxiliary array $A_2$ is employed to prevent overflow when two adjacent low quantization values $L$ are transformed by RCM. Then, the two arrays, $A_1$ and $A_2$, are added to the data to be embedded along with message bits. This principle reduces the number of data bits that can be hidden into an AMBTC image.

Type-II AMBTC-based RDH techniques [12]–[18] use the reconstructed AMBTC pixels to embed secret bits instead of DSs. Therefore, the number of bits that can be hidden in an image compressed using the AMBTC algorithm can reach hundreds of thousands.

However, the size of the stego images remains the same as that of the uncompressed images, making it impossible to leverage the reduced size of AMBTC-compressed images during transmission. In addition, it is obvious that the stego decompressed images cannot be decoded by traditional AMBTC decoders since they are already in the reconstructed form.

Generally, these analyzed Type-I and Type-II AMBTC-based RDH schemes are challenged with the following issues:

1) An embedding payload is limited.

2) The length of the quantization levels, $H$ and $L$, is expanded to embed the secret message bits. This leads to a reduction in security against detection attacks and decreases the ability to decode using the standard AMBTC decoder.

3) The structure of the stego AMBTC images is significantly degraded due to the large amount of changes applied to the AMBTC trios.

To address the aforementioned problems, a novel AMBTC-based RDH scheme is proposed with the following contributions.

1) A new and simple RDH algorithm is presented, which employs AMBTC trios to hide the secret bits.

2) A new technique is proposed to classify the AMBTC bitmaps, ensuring reversibility and optimizing the auxiliary information.

3) A mechanism to trade-off between EC and visual quality of the stego images by utilizing the two predefined thresholds. Therefore, the embedding payload can be adaptively adjusted.

4) The proposed scheme achieves a higher EC compared to the existing RDH schemes related to Type-I AMBTC based RDH.

The remainder of this paper is organized as follows. In section 2 related work, the short description of the AMBTC compression algorithm is given, and then the related AMBTC-based RDH scheme introduced by Zheng *et al.* [8] is discussed. The proposed AMBTC-based RDH scheme is given in section 3. Section 4 presents the experimental results and discussion.

## 2. Related work

In this section, the related state-of-the-art methods are reviewed. The standard AMBTC compression method is briefly explained in the following. Then, the two state-of-the-art related Type-I AMBTC-based RDH schemes, Zheng *et al.*'s method [8] and Hui *et al.*'s method [9], are presented in this section.

### *2.1. AMBTC compression technique*

The image $I$ with dimensions $HI \times WI$ is divided into $M$ consecutive blocks with size of $r \times r$:

$$M = \left\lfloor \frac{HI}{r} \right\rfloor \times \left\lfloor \frac{WI}{r} \right\rfloor \tag{1}$$

Consider any block, denoted by the element $x_j$ of the block, where $j = 1, 2, \ldots, r^2$ represents the pixel index assigned from top to bottom, left to right. The average value of the pixels in the block is calculated using the formula:

$$V = \frac{1}{r^2} \sum_{j=1}^{r^2} x_j \tag{2}$$

The bit sequence $B$ is computed based on the relationship between the pixel elements and the average value. If $x_j < V$, then $B_j = 0$, otherwise, $B_j = 1$. Let $q$ denote the number of bits '1' in $B$, the two quantities $L$ and $H$ are approximately calculated using the following formulas:

$$L = \frac{1}{r^2 - q} \sum_{B_j=0} x_j, \tag{3}$$

$$H = \frac{1}{q} \sum_{B_j=1} x_j \tag{4}$$

| | | | |
|---|---|---|---|
| 246 | 246 | 245 | 246 |
| 246 | 247 | 246 | 245 |
| 245 | 242 | 243 | 244 |
| 242 | 237 | 235 | 231 |

*(a)*

| | | | |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 |

*(b)*

| | | | |
|---|---|---|---|
| 244 | 244 | 244 | 244 |
| 244 | 244 | 244 | 244 |
| 244 | 244 | 244 | 229 |
| 229 | 244 | 229 | 229 |

*(c)*

*Fig. 1. (a) Natural image block, (b) AMBTC bitmap block, (c) Reconstructed image block.*

Figure 1(a) illustrates an example of an original image block (uncompressed image's pixels) with an average value calculated by formula (2) and the binary block $B$ determined based on $V = 242.875$. The results are shown in Figure 1(b). The values of $L$ and $H$ are calculated according to formulas (3) and (4), respectively. With values $L = 229$ and $H = 244$, the corresponding decompressed image block is shown in Figure 1(c).

### 2.2. A novel adjustable RDH method for AMBTC-compressed codes using one-to-many map

To enhance EC of RDH method, Zheng *et al.* [8] proposed a high-capacity Type-I AMBTC-based RDH scheme. In this approach, a simple one-to-many map between the used and unused bitmaps in an AMBTC compressed image is utilized based on the characteristics of bitmaps $B_i$. A bitmap generally consists of $r \times r$ bits; therefore, it has the total $2^{r\times r}$ states. If $B_i = B_j$, then these bitmaps are classified in the same class. Suppose that the number of all classes in an AMBTC compressed image is $p$, $K = 2^{r\times r} - 1 - p$ and $Z_t$ is an unused states set, $t \in \{1, 2, \ldots, K\}$. And $q$ is a set that contains the unique bitmaps which does not provide any payload. $p_{-q} = p - |q| - 1$ are the sets which have more than two bitmaps, excluding the classes that $L_i = H_i$. Thus, $k = \lfloor K/(p - |q| - 1) \rfloor$ is the unused states divided equally to each class in $p_{-q}$. Data embedding is achieved by conducting one-to-many map between $p$ used classes and $2^{r\times r} - 1 - p$ unused classes. To guarantee reversibility, the first bitmap of each class is not used in data hiding.

All unused states $\{Z_1, Z_2, \ldots, Z_k\}$ are mapping to classes in $p_{-q}$ as $T_1 = \{Z_1, Z_2, \ldots, Z_k\}$, $T_2 = \{Z_{k+1}, Z_{k+2}, \ldots, Z_{2k}\}, \ldots,$ $T_r = \{Z_{(n-1)k+1}, Z_{(n-1)k+2}, \ldots, Z_{nk}\}$, where $n = \lfloor K/k \rfloor$.

The embedding process consists of two main parts. The first part is the preprocess, and the second one is data hiding. The major purpose of the preprocess is combined by two tasks. The first task is to construct a location map with a size of $p$, denoted by $A$ to guarantee reversible recovery at the decoding phase. $A$ is then embedded to the LSBs of the quantization levels $H$. As a result, the LSBs of $H$ elements are stored in $Q$ being concatenated with the message bits to be embedded. The second task is performed on each trio block $(H_i, L_i, B_i)$, if $H_i \neq L_i$ and $B_i$ is the first bitmap in its class in $p_{-q} \cup q$, then $H_i = H_i + 1$ and swap $H_i$ and $L_i$.

The detailed data embedding algorithm is described as follows [8]:

**Input:** AMBTC compressed codes $\{(H'_i, L'_i, B_i)\}_{i=1}^{M}$, secret data $S'$.
**Output:** AMBTC stego codes $\{(H'_i, L'_i, B'_i)\}_{i=1}^{M}$

1) Read the trios $\{(H'_i, L'_i, B_i)\}_{i=1}^{N}$ sequentially.
2) For $(H'_i, L'_i, B_i)$, if $H'_i = L'_i$, 16 message bits are extracted from $S'$ and $B'_i$ is formed by replacing $B_i$ with 16 bits of data.
3) For $(H'_i, L'_i, B_i)$, if $H'_i < L'_i$, then it is recorded in the one-to-$(k + 1)$ map. Meanwhile, $B'_i$ is generated by flipping each element of $B_i$.
4) For $(H'_i, L'_i, B_i)$, if $H'_i > L'_i$, then the following three cases are performed to embed $\log_2(k + 1)$ bits into the scanned trio:

    **Case 1:** Convert secret information $S'$ from binary to $(k + 1) - ary$, where the value of $k$ is determined by Eq. (1) in [8]. Specifically, $k$ represents the number of states that each class in the set $p - |q| - 1$ can correspond to. The result plays a crucial role in determining the EC of this approach.

    **Case 2:** Suppose that $B_i$ belongs to the class $T_n$, where $n \in \{1, \cdots, p\}$. Then,

$T_n$ and the $k$ unused states $\{z_{(n-1)\times k+1},\ z_{(n-1)\times k+2}, \cdots, z_{nk}\}$ construct the mapping set of $B_i$.

**Case 3:** If the to-be-embedded bit is $s = 0$, then $B_i$ is mapped to itself; if $s = 1$, then $B_i$ is mapped to $z_{(n-1)\times k+1}$, and so on; if $s = k+1$, then $B_i$ is mapped to $z_{nk}$. Suppose that $B_i \in T_1$ and $s = 5$, and $B_i$ is replaced by $z_5$. Thus, the output is the AMBTC stego code $(H'_i, L'_i, z_5)$.

5) Repeat steps 1-4 until all bits of secret data are embedded.
6) After data hiding is completed, the stego codes are output.

The stego codes are transferred to the receiver via the Internet, and then hidden data is extracted by employing a data extracting and recovering algorithm. The proposed data extracting and recovering algorithm is as follows:

**Input:** AMBTC stego codes $\{(H'_i, L'_i, B'_i)\}_{i=1}^{M}$
**Output:** Secret data $S'$, the original AMBTC compressed codes $\{(H_i, L_i, B_i)\}_{i=1}^{M}$

1) Scan the AMBTC stego codes $\{(H'_i, L'_i, B'_i)\}_{i=1}^{M}$ sequentially according to the same order as embedding.
2) For $(H'_i, L'_i, B'_i)$, if $H'_i = L'_i$, all 16 bits of $B'_i$ are extracted as the secret data, and then, $B_i$ is set as a matrix whose elements are '1', and $H_i = H'_i$, $L_i = L'_i$.
3) For $(H'_i, L'_i, B'_i)$, if $H'_i < L'_i$, $B_i = B'_i$.
4) For $(H'_i, L'_i, B'_i)$, if $H'_i > L'_i$, Figure 2 in [8] is checked to find which range $B'_i$ belongs to. For example, if it is the same as $z_5$ of the range $\{z_1, z_2, \cdots, z_k\}$, then the secret bit is 5, and $B_i = T_1$.
5) Repeat steps 1-4 until all bits of secret data $S'$ are extracted. The extracted data $S'$ consists of two parts: one is the $Q$ with size of $p$ and the other is the secret data $S$.
6) Extract the first $p$ bits of $S'$ and replace the LSBs of $H'_i$ in the first $p$ blocks where $H'_i$ is not equal to $L'_i$.
7) For the first $p$ blocks where $H'_i$ is not equal to $L'_i$, if $H'_i > L'_i$, $H_i = H'_i - 1$, $L_i = L'_i$. If $H'_i < L'_i$, $H_i = H'_i$, $L_i = L'_i - 1$.
8) Once data extracting is completed, the original AMBTC codes are output.

The embedding payload is enhanced by utilizing all elements ($H$, $L$ and $BM$) of a trio to hide the message bits. However, due to the principle of the embedding process, identifying the class to which the considered trio belongs is required in order to extract the secret bits. This requires a set to store all used classes; thus, this set needs to be sent to the receiver or it needs to be reconstructed as well.

Moreover, an additional location map is further necessary to record cases where $H+1$ exceeds 255. This auxiliary data is essential to ensure the correctness of the original image recovery. In addition, the actual number of secret bits embedded into the AMBTC trios must be reduced by the number of bits used to store the LSBs (named as $Q$ in [8]) of the high quantization levels $H$ of the first bitmap in each class.

### *2.3. An RDH method for AMBTC compressed image without expansion inside stego format*

To increase the number of secret bits that can be embedded into the AMBTC compressed code, Hui *et al.* [9] presented the RDH method. In which, the AMBTC trios are classified into two categories: smooth and complex trios, to hide data bits. The embedding procedure is organized by three phases: pre-process, embedding for smooth trios and embedding for complex trios. The pre-process phase generates three supported arrays: $CS$, $CC$ and $CL$. $CL$ contains the smooth trios with the $H$ element, then this array is altered to use as an indicator of smoothness. If the considered trio is identified as a complex trio, then its $H$ element is modified and utilized as a smoothness indicator. The low quantity level $l$ is appended to $CL$.

The step by step operators of the data embedding for smooth trios are presented as follows:

1) For each trio in $CS$, the difference between the two quantity levels, $H$ and $L$, is calculated as $d_i = H_i - L_i$. Then, $d_i$ is encoded by looking up to the Huffman table, which is pre-generated from the values of $L$.
2) If the length of the corresponding Huffman code of $d_i$ is smaller than 8, then $8 - n^r$ message bits are concealed into $L$ of the trio, where $r$ is the size of binary Huffman code of $d_i$.

In the next stage, each pair ($p_i$,$q_i$) in $CL$ is transformed to ($p_i^{'}$,$q_i^{'}$) by applying RCM. Then, message bits are embedded into ($p_i^{'}$,$q_i^{'}$) based on the parity of this pair.

The Huffman code is applied to prevent the expansion in stego code. In other words, the size of $H$ and $L$ is maintained unchanged. However, the use of this compression code increases the complexity of implementing this scheme in real-time applications. Moreover, although the embedding payload is increased compared to related previous AMBTC-based RDH methods, the number of message bits that can be embedded is reduced due to the two extra information arrays employed [9]. The first array stores the original values of the quantization levels *H*, which are modified during data embedding to mark smooth trios. The second array records the locations of overflow issues.

## 3. Proposed scheme

In this section, the proposed adaptive RDH scheme is presented in detail. An effective RDH method is developed to embed secret data, and an extraction algorithm is proposed to retrieve the hidden bits and completely restore the original AMBTC image.

### *3.1. An adaptive RDH algorithm utilizes AMBTC bitmap classification strategy*

As given in the detailed introduction of AMBTC encoding, the compressed trio $\Psi$ ($H$, $L$, $BM$) is an output of this process. These three arrays in trios set consist of

$M$ elements, where $M$ is identified by Eq. (1). The bitmap $BM$ is a binary matrix of size $r \times r$. Thus, it can represent a decimal value from 0 to $2^{r \times r-1}$.

**Algorithm 1:** An adaptive AMBTC-based RDH Algorithm utilizes Bitmap Classification Strategy

```
Data: cover AMBTC trios $\Psi(I) = (H_i, L_i, BM_i)$, secret bits $W$,
      Quantization Value Threshold $Thr$, $noOfMsg2Embed$, Frequency Threshold $\lambda$
Result: $nOfMsgEmbedded$, stego AMBTC trios $\Psi^{'} = \{H_i^{'}, L_i^{'}, BM_i^{'}\}, LM, N$
1. Traverses through a set $\Psi$ to enumerate histogram of bitmaps, stores them in $hBM$;
2. Identify $\Omega_0 \leftarrow$ find bitmaps $BM_i$ where $H_i = L_i$;
3. Identify $\Omega_{21} \leftarrow$ the representative bitmap;
$N \leftarrow size(\Omega_{21})$;
$p \leftarrow \lceil log_2(N) \rceil$, $r_p \leftarrow r^2 - p$;
$LM \leftarrow []; t \leftarrow 0$ ;$repBM \leftarrow []$;
$i = 1; msgInx \leftarrow 0$, $t = 0$;
for i=1 to N do
    if $(H_i == L_i)$ then
        // $BM_i \in \Omega_0$
        $W_i \leftarrow W(msgInx + 1 : msgInx + r^2)$;
        $BM_i^{'} \leftarrow W_i$;
        $msgInx = msgInx + r^2$ ;
    else
        $blkVal = B2D(BM_i)$; $h \leftarrow hBM(blkVal)$;
        if $(h < \lambda || h == 1)$ then
            // $BM_i \in \Omega_1$
            $(L_i^{'}, H_i^{'})$=swap$(L_i, H_i)$; $LM(t+1) = 0; t = t + 1$;
            $BM_i^{'} = \sim BM_i$;
        else
            if $(h \geq \lambda)$ then
                if $(isempty(find(repBM, blkVal)))$ then
                    // $BM_i \notin \Omega_{21}$
                    add $BM_i$ to $repBM$ along with $blkVal$;
                    $(L_i^{'}, H_i^{'})$=swap$(H_i, L_i)$; $LM(t+1) = 1; t = t + 1$;
                    $BM_i^{'} = \sim BM_i$;
                else
                    if $((H_i > L_i) \&\& (|H_i - L_i| \leq Thr))$ then
                        $tPos$ = find($repBM$,$blkVal$);
                        $rBM$ = $repBM(tPost)$.BM;
                        $BM_i^{'}(1{:}p)$ = dec2bin($tPos$,$p$);
                        $W_i \leftarrow W(msgInx + 1 : msgInx + r_p)$, $msgInx = msgInx + r_p$;
                        $BM_i^{'}(p+1 : p+r_p) = W_i \oplus rBM(p+1 : p+r_p)$;
                    end
                end
            end
        end
    end
end
```

It is well known that the value of $BM$ elements is 0 or 1; therefore, there are existing several bitmaps which have the same $r \times r$ elements. These bitmaps are classified into a class as shown in Figure 2. Generally, there are existing classes that consist of only one bitmap (singleton classes) and others that contain more than one bitmap (multi-element classes).

According to the main principle of the proposed approach, the given message bits are concealed within the bitmaps. The singleton class cannot be used because modifying these bitmaps cannot be reversed after data extraction. To ensure the reversibility of the original bitmaps, the first bitmap (referred to as the representative bitmap) of each multi-element class is maintained. Therefore, these representative bitmaps are added to a list named *repBM*.

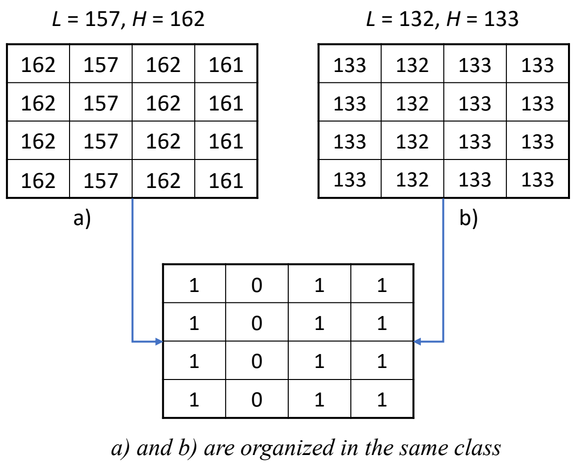


*Fig. 2. The two different pixel blocks are classified into a same class.*

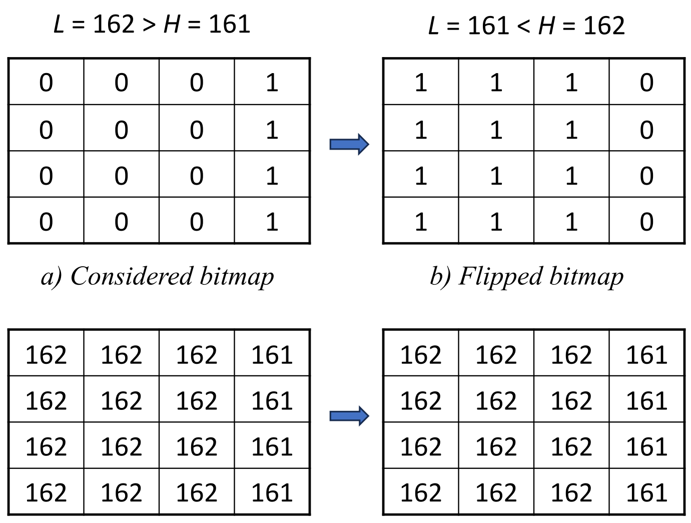
To effectively classify the bitmaps, a decimal value is computed based on the binary values and their positions within the matrix using the function B2D. In B2D, let $BM_i$ denote the input binary matrix; the corresponding decimal value is obtained by applying the following equation:

$$\begin{aligned} q_1 &= \sum_{i=0}^{7} \text{dVec}[i+1] \times 2^i \\ q_2 &= \sum_{i=0}^{7} \text{dVec}[i+9] \times 2^i \\ bd &= q_1 \times 2^8 + q_2, \end{aligned} \tag{5}$$

where *dVec* is one dimension vector transposed from $BM_i$. Set $\pi(I) = \{1, 2, \ldots, M\}$. $\Omega(I) = \{BM_i, i \in \pi(I)\}$ is a set consisting of all binary bitmaps of compressed AMBTC images.

The next process of the proposed algorithm is identifying the following sets:

1) $\Omega_0 = \{BM_i \in \Omega(I) : L_i = H_i, i \in \pi(I)\}$.
2) $\Omega_1 = \{BM_i \in \Omega(I) : H_i > L_i$ and $\forall j \in \pi(I) \setminus \{i\} \Rightarrow BM_i \neq BM_j\}$.
3) $\Omega_2 = \{BM_i \in \Omega(I) : H_i > L_i$ and $\exists j \in \pi(I) \setminus \{i\}$ such that $BM_i = BM_j\}$.



| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 |

*a) Considered bitmap*

| | | | |
|---|---|---|---|
| 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |

*b) Flipped bitmap*

| | | | |
|---|---|---|---|
| 162 | 162 | 162 | 161 |
| 162 | 162 | 162 | 161 |
| 162 | 162 | 162 | 161 |
| 162 | 162 | 162 | 161 |

| | | | |
|---|---|---|---|
| 162 | 162 | 162 | 161 |
| 162 | 162 | 162 | 161 |
| 162 | 162 | 162 | 161 |
| 162 | 162 | 162 | 161 |

*c) Constructed pixel block*

*Fig. 3. Example of marking an unused bitmap (a representative bitmap).*

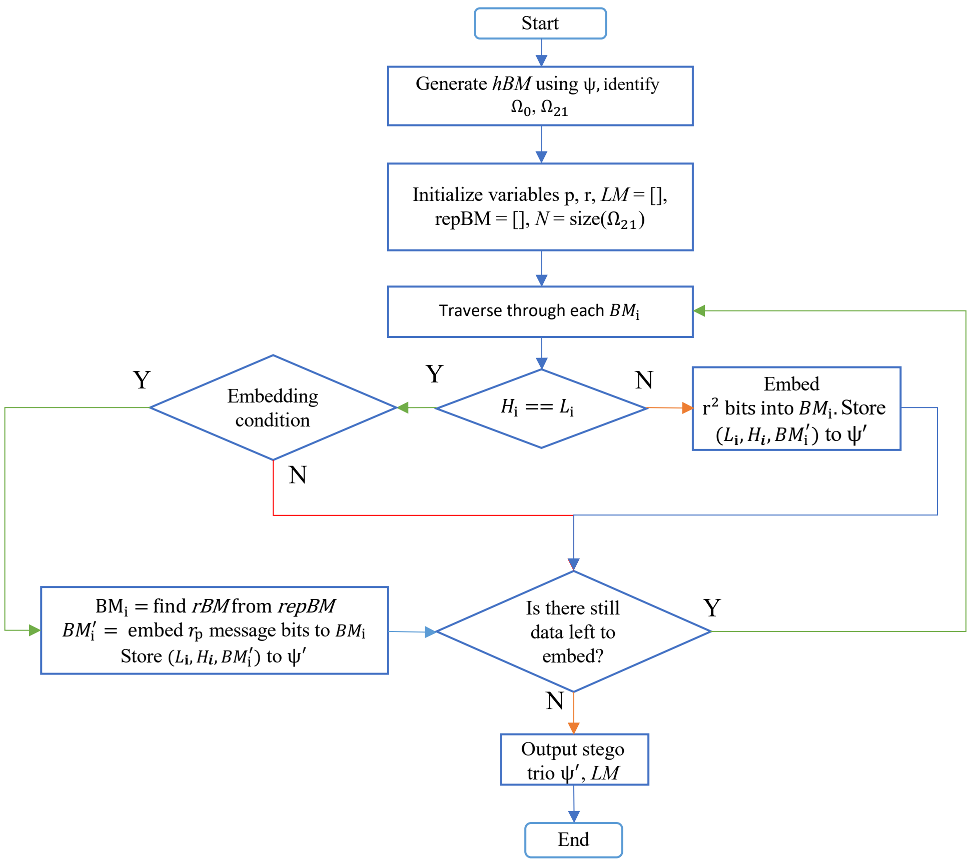

Start
Generate $hBM$ using $\psi$, identify $\Omega_0$, $\Omega_{21}$
Initialize variables p, r, $LM$ = [], repBM = [], $N$ = size($\Omega_{21}$)
Traverse through each $BM_i$
$H_i == L_i$
Y
N
Embed $r^2$ bits into $BM_i$. Store $(L_i, H_i, BM_i')$ to $\psi'$
Embedding condition
Y
N
$BM_i$ = find $rBM$ from $repBM$
$BM_i'$ = embed $r_p$ message bits to $BM_i$
Store $(L_i, H_i, BM_i')$ to $\psi'$
Is there still data left to embed?
Y
N
Output stego trio $\psi'$, $LM$
End

*Fig. 4. Flowchart of the data embedding process.*

For trios which have $H_i = L_i$, their bitmaps are stored in the set $\Omega_0$. The set $\Omega_1$ consists of bitmaps where $H_i > L_i$ and their bitmap $BM_i$ is unique. As a consequence, $\Omega_1$ is a set of single classes, while $\Omega_{22}$ is the set of to-be-embedded blocks. It is clear that $\Omega(I) = \Omega_0 \cup \Omega_1 \cup \Omega_2$ and $\Omega_0 \cap \Omega_1 = \Omega_0 \cap \Omega_2 = \Omega_1 \cap \Omega_2 = \varnothing$. Divide $\Omega_2$ into classes $P^{(1)}, P^{(2)}, \ldots, P^{(N)}$ such that:

- $\Omega_2 = P^{(1)} \cup P^{(2)} \cup \cdots \cup P^{(N)}$,
- $\forall i \neq j \in \{1, \ldots, N\}$, $P^{(i)} \cap P^{(j)} = \emptyset$,
- $\forall P^{(j)} : x, y \in P^{(j)} \implies x = y$.

Let $BM^{(t)} \in P^{(t)}$ be the bitmap that appears earliest in $\Omega(I)$ and

$$\Omega_{21} = \{BM^{(1)}, BM^{(2)}, \ldots, BM^{(N)}\}, \Omega_{22} = \Omega_2 \setminus \Omega_{21}.$$

We have:

$$p = \lceil \log_2 N \rceil, \quad r_p = r^2 - p,$$

where $p$ is the number of bits needed to represent the ordering of the employed representative bitmaps. Denote $P_t = P^{(t)} \backslash \{BM^{(t)}\}$, for $t = 1, 2, \ldots, N$. Denote $repBM$ is an array of the representative bitmaps. The major principle of the proposed RDH embedding algorithm is given as follows:

1) Each block $BM_i \in \Omega_0$ will be entirely replaced by $r \times r$ data bits.
2) The bitmap blocks belonging to $\Omega_1 \cup \Omega_{21}$ are kept intact and used to recover an original image when data extracting is finished. Reverse the values of $L_i$ and $H_i$ to differentiate the block $BM_i \in \Omega_1 \cup \Omega_{21}$ from the blocks belonging to $\Omega_{22}$. Thus, in the stego image, if $L_i' > H_i'$, then $BM_i \in \Omega_1 \cup \Omega_{21}$. To differentiate $BM_i \in \Omega_1$ or $BM_i \in \Omega_{21}$, a 0 or 1 label can be used for the block belonging to $\Omega_1$ or $\Omega_{21}$. This 0 or 1 label can be recorded in the location map (LM) or embedded in the pair $(L_i, H_i)$.
3) For each block $BM_i \in P_t$, the first $p$ bits are replaced by an equal number of bits generated from $(t, p)_2$, where the notation $(t, p)_2$ denotes $p$ bits representing the integer $t$ in base 2, and the subsequent $r^2 - p$ bits are replaced by $r_p$ secret bits.

In cases where bitmaps belonging to classes with fewer than $\lambda$ elements, they are marked by swapping the two coefficients, $H$ and $L$, and flipping all their bits, as shown in Figure 3. As demonstrated in Figure 3, although the two coefficients, $H$ and $L$, are swapped and the bits in the bitmap are flipped, the reconstructed pixel blocks remain unchanged.

To elucidate how the algorithm integrates data in AMBTC trios, a flowchart depicting the data embedding process is presented, as illustrated in Figure 4. This visualization demonstrates the impact of the $Thr$ and $\lambda$ on the visual quality of the stego images and the resulting EC.

A simple example is provided to illustrate the proposed RDH scheme and to facilitate a further understanding of its embedding method. In the case of a considered bitmap

$BM_i = [1, 0, 1, 1; 1, 0, 1, 1; 1, 0, 1, 1; 1, 0, 1, 1]$, which is extracted from the Lena image with $H = 132$ and $L = 127$, the embedding process is illustrated in Figure 5.

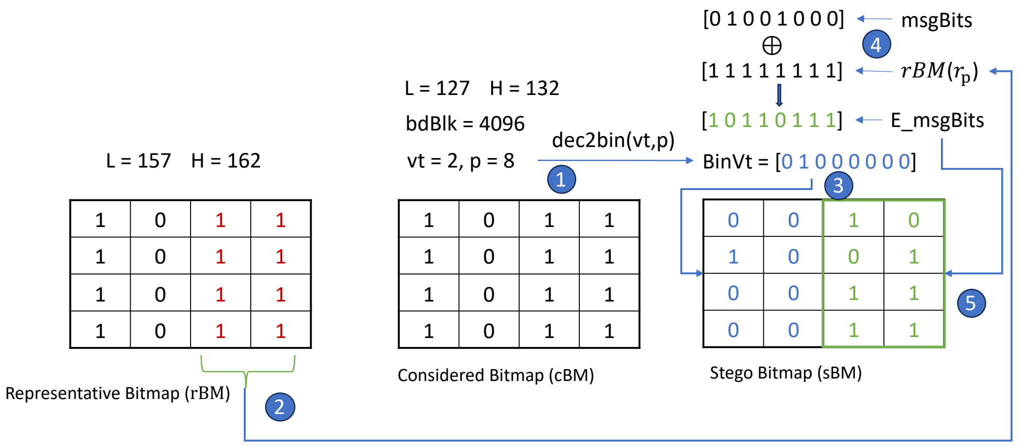
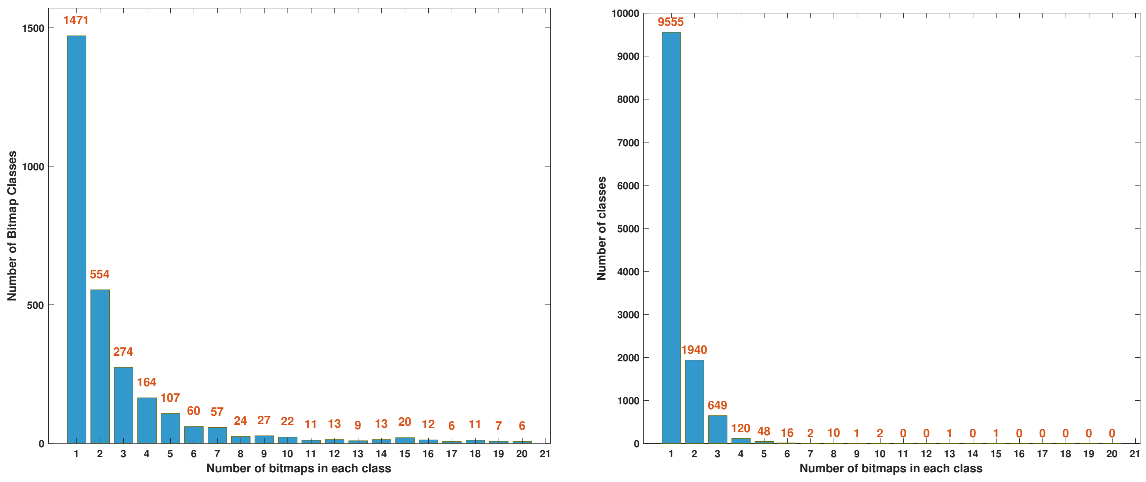


*Fig. 5. Example of embedding process.*

First, a corresponding decimal value of the $BM_i$ is calculated using Eq. (5), 4,096 is an output value, and it is stored in $bdBlk$. Next, the value of $bdBlk$ is used to identify the matched representative bitmap in the *repBM* list (a list of the representative bitmaps). A location of the corresponding representative bitmap, *vt*, is then converted into a binary vector *BinVt* = [0 1 0 0 0 0 0 0]. These bits are assigned to positions 1 through $p$ of the bitmap $BM_i$. Next, a bitwise exclusive-OR (XOR) operation is performed between the $r_p$ message bits $W_i$ and the bits $rBM(p + 1 : p + r_p)$. As a result, the bits $BM_i^{'}(p+1 : p+r_p)$ are set to the bits resulting from the XOR operation, where $BM_i^{'}$ is the $i$-th stego bitmap. Regarding the Lena image, with $N = 492$, we have $p = 8$ and $r_p = 8$. Consequently, up to 8 message bits can be embedded into an AMBTC bitmap, while the first 8 bits of the bitmap are used to store the location of the corresponding representative bitmap, ensuring successful recovery. Furthermore, the two quantization levels, $H$ and $L$, remained unchanged during the data hiding process. As a result, the degradation introduced to the stego AMBTC-compressed images, in terms of structural information and perceptual quality, is minimized.

To briefly demonstrate the transformation of binary bitmaps during the message embedding process in the proposed scheme, a statistical analysis using the Camera image was conducted. First, the number of binary bitmaps ($BMs$) in each class was enumerated. The data were then sorted in descending order by frequency, and the 20 most frequent classes are plotted in Figure 6. In the original Camera image (Figure 6(a)), there are 1,471 classes that consist of only one unique $BM$, and this number significantly increases to 9,555 in the corresponding stego image (Figure 6(b)). These 9,555 classes include the original representative $BMs$ as well as newly created ones generated during the data hiding phase.

Although the classes containing two or three $BMs$ are not used in the data hiding, the number of these classes also increased because a certain number of $BMs$ were modified during data embedding. Coincidentally, the new values of these $BMs$ (calculated using Eq. (5)) match the values of the bitmaps in the classes previously mentioned (the classes containing two or three $BMs$). This explains the increase in the number of classes containing two or three $BMs$. Consequently, classes exceeding three $BMs$ have decreased or disappeared, as they were transformed into others during data hiding, especially when those involving more than four $BMs$ were utilized in embedding.



*(a) cover camera image* *(b) stego camera image*

*Fig. 6. Frequency distribution of binary bitmap classes in the camera image before and after data embedding.*

Generally, if classes containing more than one *BM* are used to carry the given data, the EC increases when the cover image is a camera image. However, the number of $BMs$ that can be further employed to hide message bits excludes those where $H = L$. This is because these kinds of bitmaps are already used in data hiding, and each of them can carry $r^2$ message bits.

## *3.2. An message extraction and image recovery algorithm*

In the proposed approach, image recovery is concurrently performed with the message extraction process. The extraction algorithm employs the same operators utilized during the embedding phase, as outlined in Algorithm 2 and further explained in Figure 7.

For each bitmap $BM_i^{'}$, if $H_i = L_i$, the extracted message bits comprise all the bits in $BM_i^{'}$, and simultaneously, all the elements of $BM_i^{'}$ are recovered to '1'. Conversely, if ($H_i < L_i$), then $BM_i^{'}$ serves as a representative bitmap. Consequently, the two quantization levels, $H$ and $L$, are restored to their initial states. Additionally, all bits in $BM_i^{'}$ are inverted to obtain their original values.

**Algorithm 2:** An adaptive RDH Extraction Algorithm utilizes Bitmap Classification Strategy

```
Data: stego AMBTC trios $\Psi^{'}(I)(BM_i^{'}, H_i^{'}, L_i^{'}), LM, Thr, N$
Result: AMBTC trios $\Psi(I) = \{BM_i, H_i, L_i\}$, secret bits $W$
$p \leftarrow \lceil log_2(N) \rceil$, $r_p \leftarrow r^2 - p$;
$t \leftarrow 0$ ;$repBM \leftarrow []$;
$i = 1; msgInx \leftarrow 0$, $t = 0$;
for $i$=1 to $N$ do
    if $(H_i == L_i)$ then
        $W_i \leftarrow BM_i^{'}$;
        $W(msgInx + 1 : msgInx + r^2) \leftarrow W_i$;
        $msgInx = msgInx + r^2$;
        $BM_i \leftarrow ones(r, r)$;
    else
        if $(H_i < L_i)$ then
            if $(LM(t + 1) == 1)$ then
                add $BM_i^{'}$ to $repBM$;
            end
            $(L_i, H_i)$ = swap($L_i^{'}, H_i^{'},$); $t = t + 1$;
            $BM_i =\sim BM_i^{'}$;
        else
            if $((H_i > L_i)\&\&(|H_i - L_i| \leq Thr))$ then
                $tPos$ = bin2dec($BM_i^{'}(1 : p)$);
                $rBM$ = $repBM(tPost)$.BM;
                $W(msgInx + 1 : msgInx + r_p) \leftarrow BM_i^{'}(p+1:p + r_p) \oplus rBM(p + 1 : p + r_p)$;
                $msgInx = msgInx + r_p$;
                $BM_i = rBM$;
            end
        end
    end
end
```

In the case of other bitmaps that satisfy $H_i > L_i$ and $|H_i - L_i| \leq Thr$, the considered $BM_i$ is employed to carry message bits during the embedding process. Data extraction and bitmap recovery are simultaneously performed as follows:

1) **Position identification**: At first, the bits from positions 1 to $p$ of $BM_i^{'}$ are extracted and interpreted as a decimal value, denoted as *tPos*. This value is then used to identify the representative bitmap $rBM$.
2) **Message extraction**: The original secret bits are obtained by performing an XOR operation between an array of bits in *rBM* from positions $p+1$ to $p+rB$, and the corresponding bits at the same positions in $rBM$.
3) **Original bitmap recovery**: All bits of $BM_i$ are replaced by the bits in *rBM*. This is because, in the data hiding phase, the two bitmaps $BM_i$ and the corresponding *rBM* belong to the same class, and *rBM* is kept unchanged. Therefore, *rBM* serves as the original version of $BM_i$.

To better understand the detailed operation of data extraction and image recovery, a step-by-step example is presented in Figure 8.

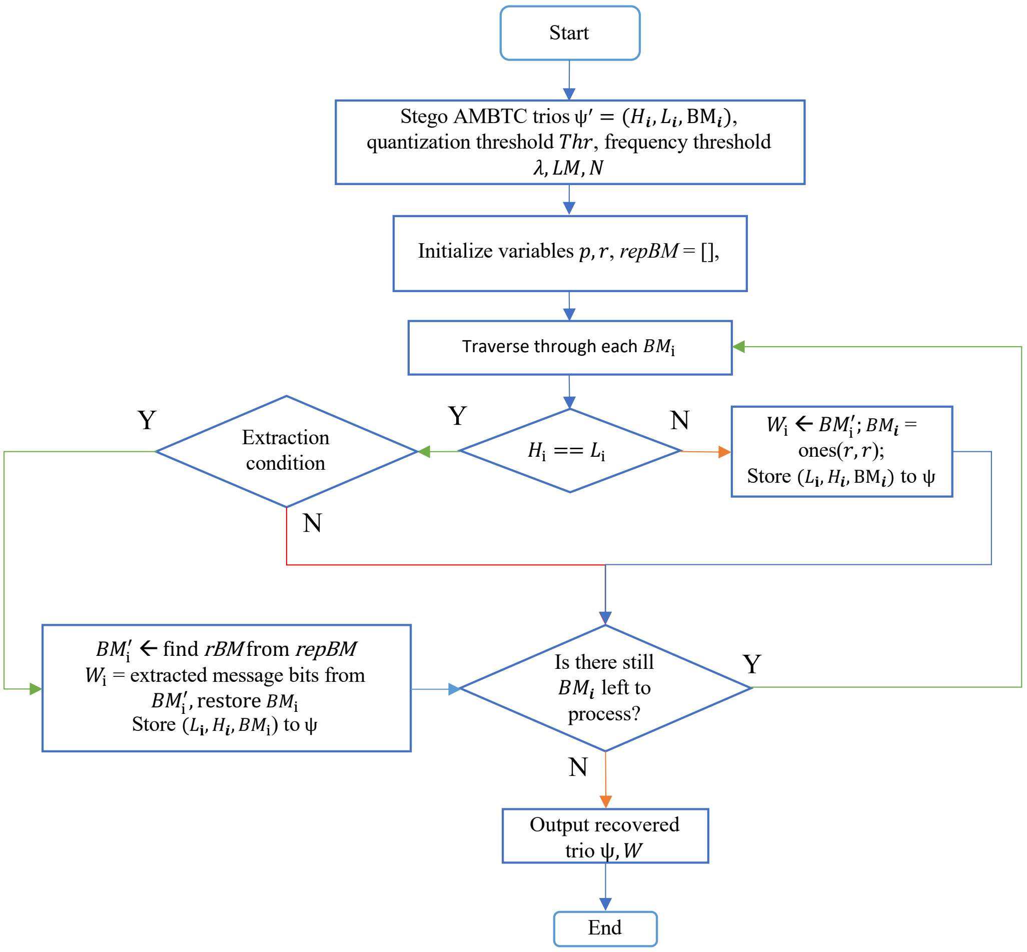
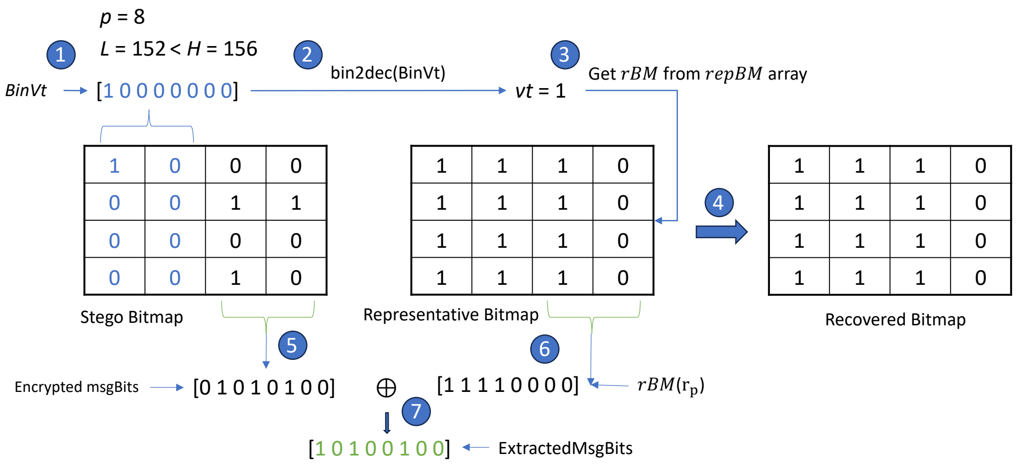

Start
Stego AMBTC trios ψ′ = ($H_i, L_i$, BM$_i$), quantization threshold $Thr$, frequency threshold $\lambda, LM, N$
Initialize variables $p, r$, $repBM$ = [],
Traverse through each $BM_i$
Extraction condition
$H_i == L_i$
$W_i \leftarrow BM_i'$; $BM_i$ = ones($r, r$); Store ($L_i, H_i$, BM$_i$) to ψ
Y
Y
N
N
$BM_i' \leftarrow$ find $rBM$ from $repBM$ $W_i$ = extracted message bits from $BM_i'$, restore $BM_i$ Store ($L_i, H_i, BM_i$) to ψ
Is there still $BM_i$ left to process?
Y
N
Output recovered trio ψ, $W$
End

Fig. 7. Data extraction and original image recovery flowchart.



Stego Bitmap:

| | | | |
|---|---|---|---|
| 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |

Representative Bitmap:

| | | | |
|---|---|---|---|
| 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |

Recovered Bitmap:

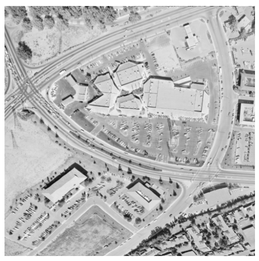
| | | | |
|---|---|---|---|
| 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 |

Fig. 8. Example of message bits extraction and image recovery.

This illustration demonstrates the operations performed on a bitmap extracted from the Lena image, where $L_i = 152 < H_i = 156$ and $p = 8$. At first, $p$ bits from the stego Bitmap ($BM_i^{'}$) are collected to generate a position *vt* of the matched representative bitmap ($rBM$). Then this $rBM$ is identified from the array *repBM* based on *vt*. The XOR operator is applied on the array of $r_p$ bits ($r_p + 1 : r_p + p$) ([1 1 1 1 0 0 0 0]) and encrypted message bits ([0 1 0 1 0 1 0 0]) to obtain the extracted message bits. Finally, the cover bitmap is its corresponding representative $BM$.

## 4. Experiment results and discussion

To demonstrate the performance of the proposed adaptive RDH scheme, experiments were conducted, and the results were obtained, analyzed, and compared with related state-of-the-art approaches. Consequently, the proposed approach and compared methods were implemented using Matlab 2022a on a laptop with an i7-10750H processor and 16 GB memory equipped.


*(a) Aerial*

*(b) Barbara*

*(c) Boat*

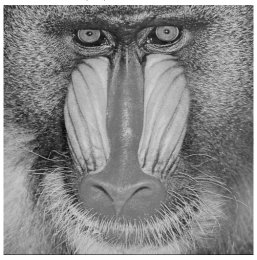
*(d) Stream*

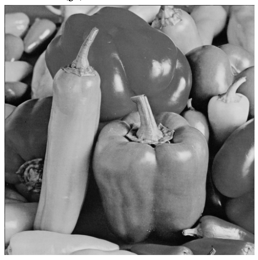
*(e) F16*

*(f) Goldhill*

*(g) Lake*

*(h) Lena*


*(i) Mandrill*


*(j) Peppers*

*(k) Tank*

*(l) Tiffany*

*Fig. 9. Test images from SIPI image library.*

In these experiments, images from the Signal and Image Processing Institute (SIPI) [19] and Kodak [20] libraries were used. Several stego images from the SIPI dataset are shown in Figure 9. Additionally, according to the principle of the proposed Type-I AMBTC-based RDH scheme and the previous relevant RDH approaches, values of the employed message bits affect to an introduced distortion in stego images. As a result, a technique for generating random bits is designed as follows. First, a message bit array $Msg$ of size $1 \times M$ is created. Then, the elements from 1 to $M/2$ of $Msg$ are assigned the value '1', while the elements from $(M/2+1)$ to $M$ are assigned the value '0'. Finally, the MATLAB function randperm is used to permute the positions of the values in *Msg* for use in the experiments.

Due to the fact that Type-I AMBTC-based RDH approaches generally utilize AMBTC trios to embed message bits, the maximum available EC is limited. Consequently, most of the latest AMBTC-based RDH schemes are Type-II, where the decoded pixels are employed to carry the given data. As a result, the EC of these approaches is higher than that of Type-I methods. Therefore, in the following experiments, the most relevant and appropriate Type-I AMBTC-based RDH approaches were implemented, even though they are not the most recently published methods, to illustrate the performance of the proposed scheme.

## 4.1. Embedding capacity and visual quality analysis

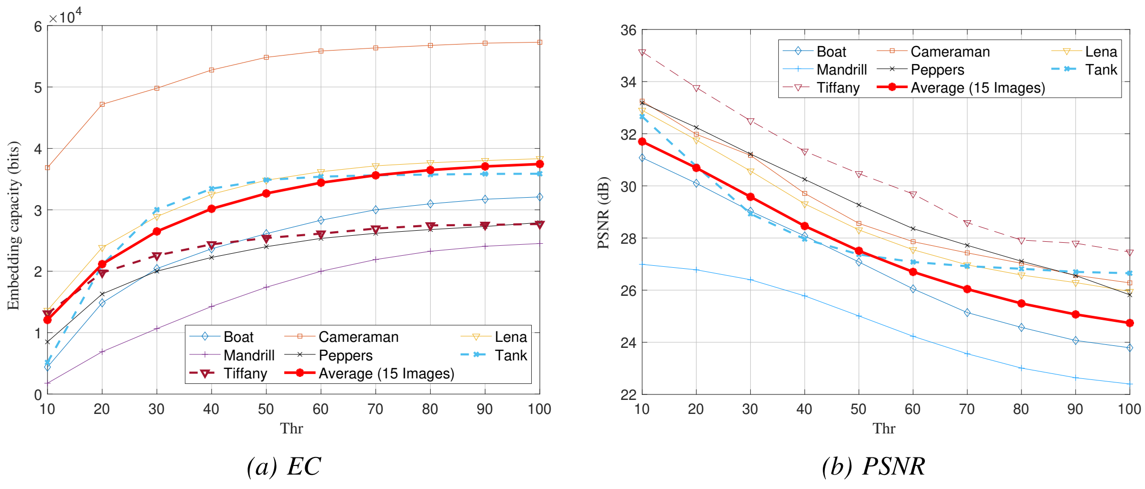


*Fig. 10. Embedding capacities (a) and PSNR (b) under different thresholds of $Thr$ for 15 test images from SIPI image library, threshold $\lambda = 4$.*

In this section, this study analyzes the maximum possible EC offered by the proposed method, as well as the corresponding visual quality. In the first experiment, the research examines the EC provided by the proposed Type-I AMBTC-based RDH scheme with various values of the threshold $Thr$ (from 10 to 100) and $\lambda = 4$ for 15 cover images from SIPI.

From Figure 10(a), it can be seen that the threshold $Thr$ contributes to different obtained EC values. The higher the value of $Thr$, the larger the obtained EC. Especially, the Goldhill image provides the highest EC, but its corresponding distortion is not the worst in comparison with other tested images as shown in Figure 10(b). The primary reason for this outcome is the usage of the embedding condition, which establishes a relationship between the absolute difference between $H_i$ and $L_i$ and the threshold value $Thr$. The binary bitmap $BM_i$ that satisfies the embedding condition, namely $|H_i - L_i| \leq Thr$, is utilized in data hiding. Consequently, the larger the threshold value $Thr$, the greater the number of binary bitmaps employed. Consequently, the obtained EC is enhanced when the value of $Thr$ is increased.

Furthermore, the threshold $Thr$ can be employed to adaptively trade off between the obtained EC and the introduced visual degradation of the stego images. For the trio *i*-th, when $|H_i - L_i| > Thr$, it is not used for data hiding. Therefore, the visual quality of stego images is enhanced, but the obtained EC can also be reduced. Consequently, if we want to achieve a high perceptual quality stego image with an appropriate EC, we can decrease the value of the threshold $Thr$.

The visual quality of stego images is an important factor in image steganography. In these experiments, the visual quality is evaluated using Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measurement (SSIM) [21], measured between the cover images and their corresponding stego AMBTC-compressed images. The PSNR metric is computed as follows:

$$\begin{aligned} PSNR &= 20 * log10(\frac{255^2}{MSE}) \\ where\ MSE &= \frac{1}{W \times H} \sum_{0}^{W-1} \sum_{0}^{H-1} \|I - S\|^2, \end{aligned} \tag{6}$$

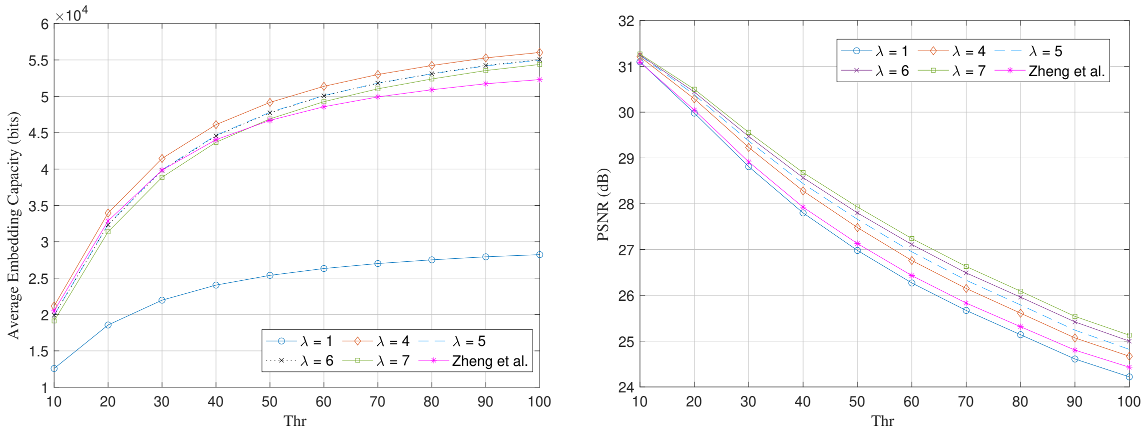
where $W$, $H$ are the width and height of an image, $I$ and $S$ are the cover and stego images, respectively.

To further examine the performance of the proposed approach in comparison with existing Type-I methods, the embedding rate (ER) is employed and defined by the following equation:

$$ER = \frac{EC}{filesize}, \tag{7}$$

where *EC* and *filesize* are the total embedded secret bits and the size of the compressed AMBTC image data, respectively. Additionally, to illustrate the performance of the proposed scheme, the size of DS of the stego images is estimated and presented in Table 1 as well.

Figure 11 presents a comparison of the EC and PSNR of the proposed scheme for various values of $\lambda$ in the range [1, 4, 5, 6, 7], as well as the method of Zheng *et al.* [8]. For the proposed AMBTC-based RDH scheme, it's obvious that the highest EC is obtained when $\lambda = 4$ and the lowest EC is corresponding to $\lambda = 1$. The Zheng *et al.*'s method provides the smallest EC in comparison with the proposed approach (when $\lambda$ is in a range of [4, 5, 6, 7]). The rationale behind this result is that, in the proposed scheme for this experiment, classes with more than 4, 5, 6, or 7 binary bitmaps are considered for data embedding. This strategy requires fewer bits to store a position of the corresponding representative bitmap of the considered BM, and produces more spaces to hide message bits. As a result, the EC of the proposed scheme is enhanced. In terms of visual quality comparison (expressed by PSNR values shown in Table 1), although the EC is highest when $\lambda = 4$, the visual quality of the stego AMBTC images produced by the proposed scheme is still better than that of Zheng *et al.*'s approach. Based on the obtained results in terms of EC and PSNR, it is observed that the value of threshold $\lambda$ influences the available embedding payload of stego images. Similar to the threshold $Thr$, $\lambda$ is utilized to select $BMs$ suitable for hiding secret bits. If a $BM_i$ belongs to a class with more than $\lambda$ distinct binary bitmaps (identified by their corresponding decimal values), it is employed for data hiding. Consequently, in a cover image, the number of classes meeting this criterion is substantial, leading to an enhancement of EC. However, the increased number of embedded message bits results in a degradation of visual quality. Generally, the classes with only one unique bitmap are avoided to be used in data hiding. Modifying these unique bitmaps leads to an ambiguous scenario that cannot recover the original bitmap. Selecting a value of $\lambda$ corresponding to a large number of classes results in an enhancement of EC.



*(a) EC* *(b) PSNR*

*Fig. 11. Average embedding capacities (a) and PSNR (b) under different thresholds of $\lambda$ for 24 test images from Kodak image library.*

To clarify the performance of the proposed Type-I AMBTC-based RDH approach compared to existing relevant methods, the methods proposed by Liu *et al.* [6] and Zheng *et al.* [8] were re-implemented. The obtained results are presented in Table 1. In the case of Liu's scheme, only Part-1 was performed because the file size of the stego-compressed image increased to over 659,000 bits for single-pass and over 664,000 bits for multi-pass. Therefore, the experiment reports a size of the DS for this method as 524,288 bits and compares only the Part-1 embedding phase in this experiment. Regarding the proposed scheme, the two employed thresholds were $Thr = 60$ and $\lambda = 4$ in this experiment.

From Table 1, it can be observed that the PSNR values of the proposed scheme are higher than those of most tested images. However, for two images, Woman and Zelda, the introduced distortions are higher than those of Zheng *et al.*'s method. The main reason for this issue is that the proposed scheme provides higher EC for these two images: 38,898 bits (proposed scheme) versus 29,961 bits (Zheng *et al.*'s method) for Zelda, and 75,514 bits (proposed scheme) versus 45,864 bits (Zheng *et al.*'s method) for the Woman image.

*Table 1. A comparison of PSNR (dB), EC (bits), DS (bits), and ER (bits) across different schemes and tested images*

| **Metric** | **Scheme** | **Lena** | **Jet** | **Barbara** | **Goldhill** | **Woman** | **Zelda** | **Milkdrop** | **Average** |
|---|---|---|---|---|---|---|---|---|---|
| PSNR (dB) | AMBTC | 33.27 | 32.01 | 31.46 | 32.89 | 38.11 | 36.75 | 36.85 | 34.48 |
| | Zheng *et al.*'s method [8] | 29.90 | 30.83 | 27.59 | 29.25 | 35.99 | 33.17 | 31.60 | 31.19 |
| | Proposed scheme | 30.57 | 30.11 | 28.80 | 29.57 | 32.34 | 31.67 | 33.68 | 30.96 |
| EC (bits) | Zheng *et al.*'s method [8] | 23,249 | 32,890 | 20,111 | 20,713 | 45,864 | 29,961 | 31,080 | 29,124 |
| | Proposed scheme | 28,905 | 25,466 | 40,062 | 26,052 | 75,514 | 38,898 | 26,430 | 37,332 |
| | Liu *et al.*'s method (Part-1) [6] | 33,758 | 37,145 | 33,626 | 34,601 | 33,285 | 33,926 | 32,493 | 34,119 |
| DS (bits) | Zheng *et al.*'s method [8] | 524,288 | 524,288 | 524,288 | 524,288 | 524,288 | 524,288 | 524,288 | 524,288 |
| | Proposed scheme | 524,288 | 524,288 | 524288 | 524,288 | 524,288 | 524,288 | 524,288 | 524,288 |
| | Liu *et al.*'s method (Part-1) [6] | 524,288 | 524,288 | 524,288 | 524,288 | 524,288 | 524,288 | 524,288 | 524,288 |
| ER (bits) | Zheng *et al.*'s method [8] | 0.444 | 0.063 | 0.038 | 0.040 | 0.087 | 0.057 | 0.059 | 0.056 |
| | Proposed scheme | 0.055 | 0.049 | 0.076 | 0.050 | 0.144 | 0.074 | 0.050 | 0.071 |
| | Liu *et al.*'s method (Part-1) [6] | 0.064 | 0.071 | 0.064 | 0.066 | 0.063 | 0.065 | 0.062 | 0.065 |

Additionally, in Table 1, the PSNR values of the stego images generated by Liu *et al.*'s method [6] are not listed because this approach expands the quantization levels, $H$ and $L$, to a length of 12 bits. Overall, the proposed scheme outperforms the compared approaches in terms of visual quality for most cases. The major reason behind its superiority is that the proposed scheme avoids modifying the LSBs of the quantization levels $H$ to mark unused binary bitmaps, as is done in Zheng *et al.*'s method. In addition, when quantization level $H = L$, the given secret bits are embedded in binary bitmap without any introduced degradation in terms of visual quality and texture characteristics.

The comparison of the proposed scheme with the two other related methods is presented in Table 1. Although the values of the two utilized thresholds, $Thr$ and $\lambda$, are 60 and 4, respectively, the proposed method outperforms the compared methods in terms of EC. This improved performance is achieved through the effective bitmap classification strategy employed in the proposed scheme. This technique reduces the number of classes, which in turn decreases the number of bits required to represent the location of the representative bitmap. Consequently, more space is created, enabling the embedding of additional message bits. From these results, it is observed that the classes with four distinct binary bitmaps ($\lambda = 4$) exhibit the highest EC in the image from the Kodak library.

The superiority of the proposed Type-I AMBTC-based RDH scheme is demonstrated by its higher performance in terms of both DS and ER. For the same DS values, a higher EC contributes to the improved ER achieved by the proposed scheme.

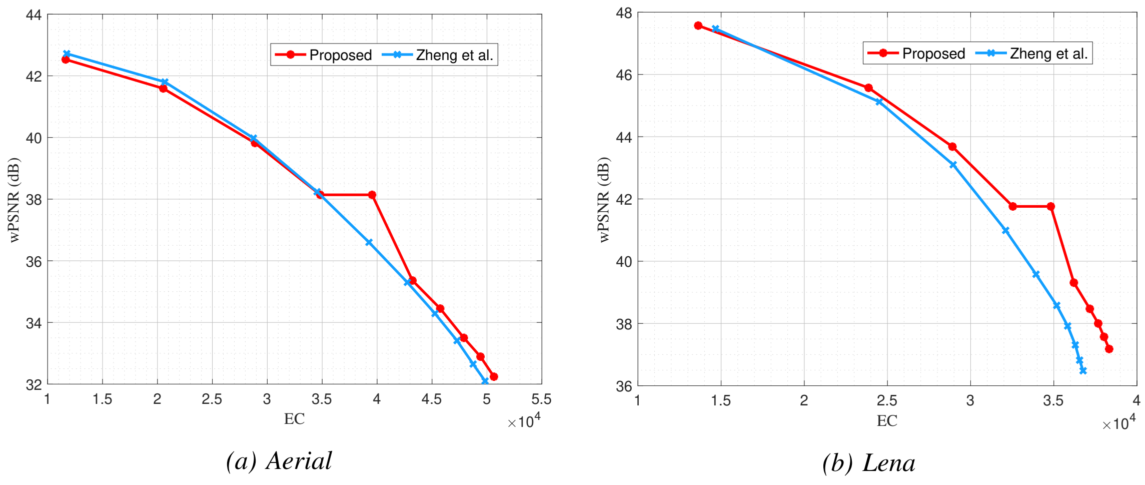


*Fig. 12. A comparison of the texture characteristics of stego images generated by the proposed scheme and those generated by Zheng et al.'s method.*

In addition, the experiment on weighted PSNR measurement is conducted to further examine the degradation in terms of texture characteristics of the stego images created by the proposed scheme and the state-of-the-art methods. The metric weighted PSNR (wPSNR) is an improved version of PSNR. It considers the properties of the Human Visual System by employing a noise visibility function (NVF) [22] and is defined as in the following equation.

$$\text{wPSNR} = 10 \times \log_{10} \frac{255 \times 255}{\text{MSE} \times \text{NVF}}, \tag{8}$$

NVF denotes the noise visibility function, which ranges from 0 (completely complex area) to 1 (completely smooth area).

NVF can be calculated using Eq. (9).

$$\text{NVF} = \text{NORM}\left(\frac{1}{1+\sigma^2_{\text{BLOCK}}}\right), \tag{9}$$

where $\sigma$ is the standard deviation and NORM function is used to normalize the obtained value to either 0 or 1.

In this experiment, message bits are embedded into the test images using the threshold variable $Thr$, which takes values ranging from 10 to 100. The obtained wPSNR values corresponding to the EC for each value of $Thr$ are shown in Figure 12. For both approaches, the proposed method and Zheng's method, the EC increases rapidly when the $Thr$ values range from 10 to 70, while it gradually decreases for values between 80 and 100. The main reason for this characteristic is that the number of AMBTC trios satisfying the condition $|H_i - L_i| \leq Thr$ depends on the texture features of the tested images. It can be observed that the texture distortion of the stego images generated by the proposed scheme is lower than that of Zheng *et al.*'s approach. This indicates that the embedding process of the proposed scheme causes less texture degradation in stego images compared to Zheng *et al.*'s method, even when more message bits are hidden in the proposed scheme.

## 4.2. Structural degradation analysis

In the AMBTC images decompression, reconstructed pixels are set to either a high quantization level *H* or a low quantization level *L* based on their corresponding bits in the bitmap. This process introduces noticeable structural distortion in stego images compared with the uncompressed images. In this experiment, the structural degradation is evaluated using the SSIM metric, which measures the similarity between the uncompressed images and the stego images. A value of 1.0 indicates that the given message is embedded into the stego image without any structural distortion. The SSIM equation is expressed as follows:

$$SSIM(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma}, \tag{10}$$

where $l, c, s$ are luminance, contrast, and structure and exponents for these terms, specified as a 3-element vector of non-negative numbers of the form $[\alpha, \beta, \gamma]$ [21]. The SSIM values, calculated from the original uncompressed images and their corresponding stego-reconstructed AMBTC images (created from the AMBTC compressed codes with embedded data), are presented in Table 2. The presented SSIM values of the AMBTC compressed images prove that the structure of the tested images is affected by the compression process. It can be seen that the proposed scheme produces stego AMBTC compressed images that are superior to those of the related state-of-the-art methods in terms of structural information.

The minor structural distortion introduced in the stego AMBTC images generated by the proposed scheme is accomplished through the application of the following operators:

1) A bitmap not used in data embedding is marked by first swapping the values of the two quantization levels, $H$ and $L$. Then, all its bits are flipped, with 0 s becoming 1 s and vice versa. This principle does not cause degradation in terms of structural information as presented in the example in Figure 3.
2) A LM is also employed to record the unused bitmaps and representative bitmaps.
3) The threshold $Thr$ is employed as a bitmap selector, as presented in the data hiding algorithm. Therefore, the unused bitmaps are marked without any degradation introduced in a stego image.

*Table 2. Comparison between the proposed Type-I AMBTC-based RDH scheme and the existing methods in terms of structural information*

| Image | Method | | | | |
|---|---|---|---|---|---|
| | AMBTC | Proposed | Zheng *et al.*'s [8] | Hui *et al.*'s [9] | Liu *et al.*'s Part-1 [6] |
| Aerial | 0.9117 | 0.6874 | 0.6714 | 0.6044 | 0.0463 |
| Barbara | 0.9330 | 0.7125 | 0.6842 | 0.4484 | 0.1045 |
| Boat | 0.9212 | 0.7654 | 0.7240 | 0.3957 | 0.0907 |
| Cameraman | 0.9522 | 0.7688 | 0.7350 | 0.2015 | 0.1088 |
| Couple | 0.9346 | 0.7003 | 0.6801 | 0.3959 | 0.0975 |
| F16 | 0.9505 | 0.8333 | 0.8121 | 0.3288 | 0.0592 |
| Goldhill | 0.9203 | 0.7284 | 0.6886 | 0.3713 | 0.1101 |
| Lake | 0.9179 | 0.8099 | 0.7714 | 0.4102 | 0.1213 |
| Lena | 0.9413 | 0.8159 | 0.7833 | 0.2629 | 0.0934 |
| Mandrill | 0.8869 | 0.7567 | 0.6978 | 0.6760 | 0.1422 |
| Peppers | 0.9349 | 0.8416 | 0.8058 | 0.2391 | 0.0867 |
| Splash | 0.9557 | 0.8752 | 0.8511 | 0.1138 | 0.0804 |
| Stream&Bridge | 0.8968 | 0.7178 | 0.6717 | 0.6850 | 0.1532 |
| Tank | 0.9024 | 0.6970 | 0.6470 | 0.4286 | 0.0947 |
| Tiffany | 0.9474 | 0.8379 | 0.8032 | 0.2781 | 0 |
| **Average** | **0.9271** | **0.7699** | **0.7351** | **0.3893** | **0.0926** |

In Zheng *et al.*'s method, besides the swapping applied to the two quantization levels of the unused bitmaps, the element $H$ is also incremented by 1 to record the first bitmap in a class. This modification results in a reduction in the structural quality of the stego AMBTC images produced by this method. In the other two compared methods, Hui *et al.*'s method and Liu *et al.*'s method, a significant amount of modification is applied to the quantization levels during data embedding. Consequently, the SSIM values of the stego AMBTC images generated by these methods are notably low. To further evaluate the structural degradation of the stego images generated by the proposed scheme and the two compared methods, Zheng *et al.*'s [8] and Hui *et al.*'s [9], the following experiment on 2,000 images which are randomly selected from BOWS2 [23], is performed. In this experiment, all tested images are accompanied by their corresponding maximum EC embedded.

Table 3 presents the average SSIM values for the original uncompressed images, their corresponding reconstructed AMBTC images, the stego images generated by the proposed scheme, the Zheng *et al.*'s approach, and the method proposed by Hui *et al.*

These results further demonstrate that the proposed method outperforms the others in terms of lower degradation of image structure.

*Table 3. A comparative analysis of the proposed Type-I AMBTC-based RDH scheme and existing methods is performed using 2,000 randomly selected BOWS2 images to assess structural information.*

| | Method | | | |
|---|---|---|---|---|
| | AMBTC | Proposed | Zheng *et al.*'s [8] | Hui *et al.*'s [9] |
| **Average SSIM** | **0.9519** | **0.8168** | **0.7763** | **0.2339** |

In general, Type-II AMBTC-based RDH approaches initially decompress AMBTC compressed code into pixels, subsequently embedding message bits into these constructed pixels. This strategy results in a degradation of structural information within stego images compared to the published Type-I AMBTC-based RDH schemes. The primary reason for this degradation is that message bits are directly concealed into pixels, and a significant magnitude of changed value may occur since Type-II methods tend to embed more data into pixels. This modification presents challenges in message exaction and original image recovery when the quantization levels ($H$ and $L$) are changed. Identifying which pixel's value corresponds to $H$ or $L$ becomes difficult. Additionally, transmitting the decompressed data embedded AMBTC images lacks the advantage of a small required transmission bandwidth, as the standard AMBTC compressed codes have a limited bandwidth. Consequently, the size of transmitted reconstructed stego images is equivalent to that of the corresponding uncompressed original images.

## *4.3. Auxiliary data analysis*

In this section, the usage of auxiliary data is discussed, specifically the location map, which is generally used in AMBTC-based RDH schemes to ensure reversibility. Figure 13 illustrates the comparison of the location map's size introduced by the three Type-I AMBTC-based RDH approaches: Zheng *et al.*'s method, Hui *et al.*'s method, and the proposed scheme.

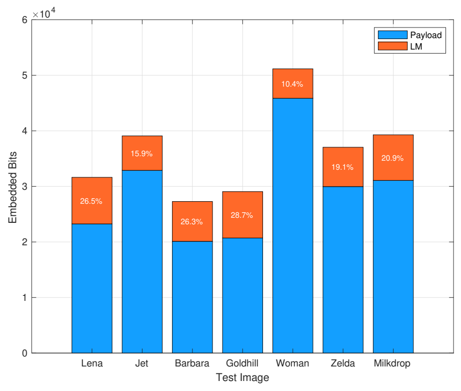
The percentage shown in these figures is calculated as follows:

$$LMPer = \frac{LMSize}{LMSize + PayloadSize}, \tag{11}$$

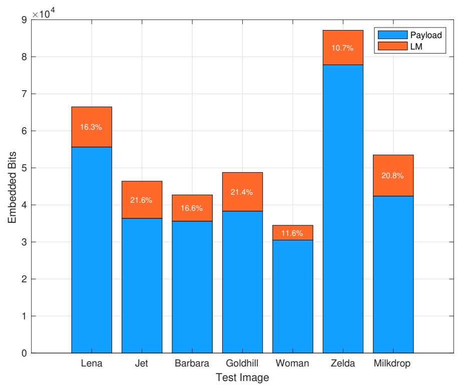
where $LMSize$ is the size of the used LM and $PayloadSize$ presents a number of the hidden message bits, respectively. As can be seen, the percentages of LMs produced by Zheng *et al.*'s method (Figure 13(a)) and Hui *et al.*'s method (Figure 13(c)) for most of the tested images are generally larger, indicating that more space is used to store auxiliary data rather than the payload itself.

LM is commonly required in all RDH schemes for the following purposes:

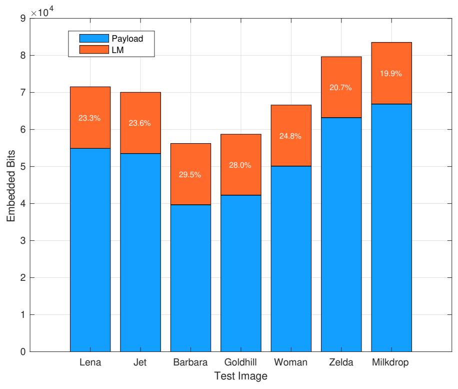
1) Record the locations where overflow or underflow issues occur to ensure the reversibility of cover images. This is because the several embedding methods employ adding/subtraction to embed message bits into the AMBTC compressed code.
2) Marking or recording the positions of the unused or representative bitmaps in AMBTC-based RDH schemes, which employ the bitmap classification technique to embed the given message bits.



*(a) Zheng et al.'s [8]*



*(b) Proposed approach*



*(c) Hui et al.'s [9]*

*Fig. 13. Comparison of LM sizes among Zheng et al.'s method (a), the proposed approach (b), and Hui et al.'s method (c).*

In the AMBTC-based RDH approach proposed by Zheng *et al.* [8], the LM is created with a size of $p$ (the number of unused bitmaps) to identify the first bitmap in a specific class. Consequently, the larger the number of classes in an AMBTC-compressed image, the larger the LM becomes. Moreover, if the LM is also used to record the location when an overflow occurs due to $H_i = H_i + 1$, the LM is expanded further. In contrast, as mentioned earlier, the scheme introduced by Hui *et al.* [9] utilizes two auxiliary arrays during the data hiding process. The first array stores the original LSBs values of the quantization levels $H$, which are modified during data embedding to mark smooth trios. Consequently, the size of this array becomes large if the cover image contains many smooth regions. The second array records the locations of overflow issues. As a result, if many overflow cases arise, the size of the second array increases.

Unlike these Type-I AMBTC-based RDH approaches, the proposed scheme generates a reasonable LM when this auxiliary data is used to identify unused bitmaps and representative bitmaps. Furthermore, the utilization of the threshold $qThr$ further reduces the magnitude of auxiliary information LM in the proposed Type-I AMBTC-based RDH scheme.

### *4.4. A Comprehensive analysis of computational complexity*

In the proposed adaptive RDH algorithm, two primary processes are executed. The initial process generates a histogram of binary bitmaps, which is stored in $hBM$. Subsequently, a data hiding process is implemented to embed message bits into the bitmap using the bit replacement method. Typically, the AMBTC compressed code consists of $N$ trios, where $N$ is the number of consecutive blocks determined by performing Eq. 1 with $HI$ and $WI$, which represent the number of pixels in rows and columns of an image, respectively. The magnitude of the block is identified by $r$, typically set to 2, 4, or 8 based on the required transmission bandwidth. Each trio is constructed from two quantization levels, $H$ and $L$, and the binary bitmap $BM$. Consequently, the two major processes work on $N$ trios sequentially.

During the histogram generation process, each trio contains $r^2$ bits. Therefore, calculating the decimal value of each trio can be considered to have an $\mathcal{O}(1)$ computational complexity. Meanwhile, the $r^2 - p$ bits ($p$ remaining bits are used to store the location of the representative $BM$) of the BMs are replaced by the same number of data bits, and $r$ is a small number. Consequently, the computational complexity of the data embedding is also $\mathcal{O}(1)$. Overall, the required computational complexity of the proposed adaptive RDH algorithm is $\mathcal{O}(N)$.

Similarly, the two preceding related Type-I AMBTC-based RDH approaches, introduced by Zheng *et al.* and Hui *et al.*, also exhibit the same computational complexity, $\mathcal{O}(N)$, when they also operate with $N$ AMBTC triples. However, in the Zheng *et al.*'s approach, the reconstruction of the mapping set is required, resulting in a proportional increase in the memory resource needed for its storage.

## 5. Conclusion

In this paper, a novel Type-I AMBTC-based RDH scheme is presented, where the AMBTC compressed code is utilized for data hiding instead of the reconstructed pixels. Therefore, the stego DS can be decompressed by standard AMBTC decoders. Furthermore, the proposed bitmap classification technique is developed to enhance the embedding payload and visual quality of the stego images. In addition, we can adaptively select the compressed trios to embed the message bits, and the number of classes can be organized using predefined thresholds. This strategy enables us to effectively balance the required EC with the appropriate perceptual quality of the stego images. Experimental results demonstrate that the proposed scheme achieves higher EC and better visual quality compared to existing Type-I AMBTC-based RDH approaches. However, a shortcoming of the proposed scheme is the requirement for auxiliary information, such as a location map, to record the positions of unused bitmaps during the data hiding phase. Therefore, future work will focus on developing new techniques to further minimize the size of the location map.

## Acknowledgement

The author appreciates the assistance of our colleagues and the financial support provided by Vietnam National University, Hanoi, School of Interdisciplinary Sciences and Arts.

## References

[1] M. Xiao, X. Li, W. Lu, and Y. Zhao, "Histogram shifting based reversible data hiding with multiple expansion bin pairs," *Displays*, vol. 83, Jul. 2024. DOI: 10.1016/j.displa.2024.102674

[2] W. He, L. Wang, Y. Wang, Z. Cai, and G. Xiong, "Reversible data hiding using morphology based pixel classification," *Expert Systems with Applications*, vol. 250, 2024. DOI: 10.1016/j.eswa.2024.123844

[3] S. S. Chen, C. C. Chang, and J. H. Horng, "Reversible data hiding scheme using prediction neural network and adaptive modulation mapping," *Multimedia Tools and Applications*, vol. 84, pp. 665–6686, 2024. DOI: 10.1007/s11042-024-19162-3

[4] G. Melendez-Melendez, A. Morales-Reyes, and R. Cumplido, "An adaptive pixel value ordering based reversible data hiding scheme for images," *Expert Systems with Applications*, vol. 232, 2023. DOI: 10.1016/j.eswa.2023.120809

[5] N. Kumar, R. Kumar, A. Malik, S. Singh, and K. H. Jung, "Reversible data hiding with high visual quality using pairwise PVO and PEE," *Multimedia Tools and Applications*, vol. 82, no. 20, pp. 30 733–30 758, 2023. DOI: 10.1007/s11042-023-14867-3

[6] J. C. Liu, Y. Lin, C. C. Chang, and C. C. Chang, "A side match oriented data hiding based on absolute moment block truncation encoding mechanism with reversibility," *Multimedia Tools and Applications*, vol. 84, pp. 18 395–18 417, 2024. DOI: 10.1007/s11042-024-19752-1

[7] C. C. Chang, X. Wang, and C. C. Lin, "An efficient dual prediction based reversible data hiding and reduced code method for AMBTC," *Multimedia Tools and Applications*, vol. 80, no. 24, pp. 33 157–33 176, 2021. DOI: 10.1007/s11042-021-11048-y

[8] W. Zheng, C. C. Chang, and S. Weng, "A novel adjustable RDH method for AMBTC-compressed codes using one-to-many map," *IEEE Access*, vol. 8, pp. 13 105–13 118, 2020. DOI: 10.1109/ACCESS.2020.2963891

[9] Z. Hui and Q. Zhou, "A reversible data hiding method for AMBTC compressed image without expansion inside stego format," *Transactions on Internet and Information Systems*, vol. 14, no. 11, pp. 4443–4462, 2020. DOI: 10.3837/tiis.2020.11.011

[10] C. C. Chang, T. S. Chen, Y. K. Wang, and Y. Liu, "A reversible data hiding scheme based on absolute moment block truncation coding compression using exclusive OR operator," *Multimedia Tools and Applications*, vol. 77, no. 7, pp. 9039–9053, 2017. DOI: 10.1007/s11042-017-4800-0

[11] Z. Yin, X. Niu, X. Zhang, J. Tang, and B. Luo, "Reversible data hiding in encrypted AMBTC images," *Multimedia Tools and Applications*, vol. 77, no. 14, pp. 18 067–18 083, 2017. DOI: 10.1007/s11042-017-4957-6

[12] T. S. Nguyen, C. C. Chang, and C. C. Lin, "High capacity reversible data hiding scheme based on AMBTC for encrypted images," *Journal of Internet Technology*, vol. 23, no. 2, pp. 255–266, 2022. DOI: 10.53106/160792642022032302006

[13] X. Zhou, G. Yang, W. Hong, K. S. Chen, and T. S. Chen, "An adaptive data hiding for AMBTC compressed images with recoverability using bound shifting technique," *Multimedia Tools and Applications*, vol. 82, no. 10, pp. 15 593–15 612, 2022. DOI: 10.1007/s11042-022-13961-2

[14] R. Bhardwaj, "A high payload reversible data hiding algorithm for homomorphic encrypted absolute moment block truncation coding compressed images," *Multimedia Tools and Applications*, vol. 80, pp. 26 161–26 179, 2021. DOI: 10.1007/s11042-021-10722-5

[15] C. Kim, D. Shin, and C. N. Yang, "High capacity data hiding with absolute moment block truncation coding image based on interpolation," *Mathematical Biosciences and Engineering*, vol. 17, no. 1, pp. 160–178, 2020. DOI: 10.3934/mbe.2020009

[16] H. Y. Wang, H. J. Lin, X. Y. Gao, W. H. Cheng, and Y. Y. Chen, "Reversible AMBTC-based data hiding with security improvement by chaotic encryption," *IEEE Access*, vol. 7, pp. 38 337–38 347, 2019. DOI: 10.1109/ACCESS.2019.2906500

[17] W. Hong, Y. B. Ma, H. C. Wu, and T. S. Chen, "An efficient reversible data hiding method for AMBTC compressed images," *Multimedia Tools and Applications*, vol. 76, no. 4, pp. 5441–5460, 2016. DOI: 10.1007/s11042-016-4032-8

[18] 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 and Applications*, vol. 74, no. 11, pp. 3823–3842, 2013. DOI: 10.1007/s11042-013-1801-5

[19] A. G. Weber, "The USC-SIPI image database," *Signal and Image Processing Institute of the University of Southern California*, 1977. [Online]. Available: https://sipi.usc.edu/database/

[20] Kodak, "Kodak lossless true color image suite," 1999. [Online]. Available: https://r0k.us/graphics/kodak/

[21] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004. DOI: 10.1109/TIP.2003.819861

[22] S. Voloshynovskiy, A. Herrigel, N. Baumgaertner, and T. Pun, "A stochastic approach to content adaptive digital image watermarking," in *Lecture Notes in Computer Science* 2000, pp. 211–236. DOI: 10.1007/10719724_16

[23] T. P. Patrick Bas, Tomas Filler, "BOWS2 image database," April 2008. [Online]. Available: https://data.mendeley.com/datasets/kb3ngxfmjw/1

*Manuscript received 9-3-2025; Accepted 15-06-2025.*

■

**Duc Tuan Nguyen** received the Ph.D. degree in Computer Science from Khon Kaen University (KKU), Khon Kaen, Thailand, in 2016. He received the M.Sc. degree from Le Quy Don Technical University, Vietnam, in 2008. He is currently a lecturer at Vietnam National University, Hanoi, School of Interdisciplinary Sciences and Arts. His current research interests include cryptography, steganography, reversible data hiding, blockchain, and machine learning. E-mail: tuannguyenduc@vnu.edu.vn.

# LƯỢC ĐỒ GIẤU TIN CÓ THỂ HỒI PHỤC SỬ DỤNG PHƯƠNG PHÁP PHÂN LỚP BITMAP CHO ẢNH NÉN AMBTC

*Nguyễn Đức Tuấn*

**Tóm tắt**

Giấu tin có thể hồi phục (*Reversible Data Hiding* - RDH) đang nhận được nhiều sự quan tâm từ các nhà khoa học. Ảnh nén AMBTC (*Absolute Moment Block Truncation Coding*) là một trong những đối tượng mang tin được sử dụng phổ biến trong các lược đồ RDH nhờ sự đơn giản và cung cấp dung lượng giấu cao. Tuy nhiên, hiện nay hầu hết các lược đồ RDH sử dụng AMBTC đều tạo ra các ảnh mang tin mà không thể giải mã bởi các bộ giải mã AMBTC tiêu chuẩn bởi vì các lược đồ này sử dụng các điểm ảnh tái tạo để nhúng tin. Ngược lại, có một số lược đồ tạo ra các ảnh mang tin tương thích với bộ giải mã AMBTC tiêu chuẩn nhưng dung lượng giấu đạt được là hạn chế. Để giải quyết vấn đề này, một lược đồ giấu tin dung lượng cao sử dụng ảnh AMBTC tương thích với bộ giải mã AMBTC tiêu chuẩn được giới thiệu trong bài báo này. Trong lược đồ này, một kỹ thuật hiệu quả để phân lớp các bitmap được phát triển và sử dụng để cung cấp khả năng hồi phục và nâng cao dung lượng giấu. Hơn nữa, lược đồ đề xuất còn cung cấp khả năng cân bằng linh hoạt giữa dung lượng giấu và chất lượng cảm quan của ảnh mang tin bằng cách sử dụng hai ngưỡng được định nghĩa trước. Các kết quả thực nghiệm cho thấy lược đồ đề xuất vượt trội hơn so với các phương pháp liên quan về cả dung lượng giấu và chất lượng cảm quan của ảnh mang tin.

**Từ khóa**

Giấu tin có thể hồi phục; khoá an ninh; thích ứng; sơ đồ vị trí; phân lớp bitmap; ảnh nén AMBTC.