Indicator Elimination for Locally Adaptive Scheme Using Data Hiding Technique

Hon-Hang Chang1, Yung-Chen Chou2, and Timothy K. Shih1
[1] Department of Computer Science and Information Engineering,
National Central University, Taoyuan 32001, Taiwan, R.O.C.
[e-mail: {sicachang, timothykshih}@gmail.com]
[2] Department of Computer Science and Information Engineering,
Asia University, Taichung 41354, Taiwan, R.O.C.
[e-mail: yungchen@gmail.com]
*Corresponding author: Yung-Chen Chou

Received May 19, 2014; revised October 11, 2014; accepted November 9, 2014; published December 31, 2014

Abstract

Image compression is a popular research issue that focuses on the problems of reducing the size of multimedia files. Vector Quantization (VQ) is a well-known lossy compression method which can significantly reduce the size of a digital image while maintaining acceptable visual quality. A locally adaptive scheme (LAS) was proposed to improve the compression rate of VQ in 1997. However, a LAS needs extra indicators to indicate the sources, consequently the compression rate of LAS will be affected. In this paper, we propose a novel method to eliminate the LAS indicators and so improve the compression rate. The proposed method uses the concept of data hiding to conceal the indicators, thus further improving the compression rate of LAS. From experimental results, it is clearly demonstrated that the proposed method can actually eliminate the extra indicators while successfully improving the compression rate of the LAS.

Keywords: Locally adaptive scheme, image compression, vector quantization, data hiding technique

A preliminary version of this paper appeared in IEEE ICC 2009, June 14-18, Dresden, Germany. This version includes a concrete analysis and supporting implementation results on MICAz sensor nodes. This research was supported by a research grant from the IT R&D program of MKE/IITA, the Korean government [2005-Y-001-04, Development of Next Generation Security Technology]. We express our thanks to Dr. Richard Berke who checked our manuscript.

http://dx.doi.org/10.3837/tiis.2014.12.022

1. Introduction

Modern computer systems become ever more powerful due to advances in the design and development of all aspects of information technology such as, the increasing size of mass storage, and the high speed of processing and transmission. Another advance is that multimedia data is easier to create than before and thus the large size of multimedia becomes the main burden for using, processing, and delivering the multimedia data. Thus, finding ways to effectively reduce the size of multimedia has become an important research issue in recent years. Multimedia includes various representations such as image, audio, video, text, etc. Because of the sensitivity of the human senses and the fact that multimedia consists of a large number of redundant spaces, multimedia data can be reduced, yet without losing acceptable quality.

Images are one of the most popular forms of multimedia in common usage. An image can be created by using computer software, cameras, scanners, etc. “Lossy” and “lossless” are the two data encoding methods in image compression field, where these techniques are used to reduce amount of data of an image, audio or other related multimedia. [1] [2] [3] The lossless image compression is more concerned about the visual quality of the compressed image rather than the performance of the compression rate. In other words, the decompressed image must be the same as the original image.

Lossy image compression is also an important technique for reducing the size of the image file and many lossy image compression techniques have been presented in recent years, for instance JPEG [4], vector quantization (VQ) [5], etc. For lossy image compression, a reduction in image size is more important than maintaining the visual quality. Lossy image compression techniques allow the decompressed image to contain tiny distortions that, due to the sensitivity of the human eye, are hard to distinguish.

The concept of VQ is to divide an image into non-overlapping blocks and compress these blocks with a pre-trained codebook. Here, the pre-trained codebook is trained by using any codebook algorithm (e.g. the LBG algorithm [6]). The compression rate depends on the size of the codebook and the size of codeword. For instance, if a grayscale image sized 256 × 256 is compressed by VQ with a 256 codeword codebook and a codeword size of 16 pixels, then the compression rate is (((256 × 256) / 16) × 8) / (256 × 256 × 8) = 0.0625.

Side match vector quantization (SMVQ) [7] is one of the compression methods to further improve the visual quality and the compression rate of VQ. Because of the characteristics of the natural image, a local area of an image has similar pixel distribution; SMVQ uses reconstructed neighboring blocks to predict the possible codewords for encoding a block.

Search-order coding (SOC) [8] was first proposed in 1996. The main idea of SOC is to encode a block by referring to a block which has the same VQ index value as the current encoding block. According to SOC design, if a current encoding block has a neighboring block with the same value (as the current encoding block), then the current block is encoded by using the search-order number. In order to avoid ambiguities, extra indicators are needed to record the source of the encoding codewords.

A locally adaptive scheme (LAS) [2] is also an image compression method that is based on the concept of local adaptively [9] [10]. The main idea of a LAS is to divide a VQ index table into non-overlapping blocks and, using a history list with “moving to the front” strategy, to improve the performance in terms of compression rate. The LAS method requires indicators to indicate the sources of encoding codeword.

This paper presents an indicator elimination method to improve the performance of the LAS method in terms of compression rate, because the indicators of LAS influence compression rate improvement.

We were inspired by a data hiding technique to design an effective compression method to eliminate the indicators of LAS; the purpose of the proposed method is to improve the compression rate of traditional LAS. Data hiding techniques [11-21] use a cover media to carry secret data and deliver it over a public computer network. The method proposed in this paper integrates a dissimilar pairing concept and side match vector quantization concept to conceal indicators of LAS into compression code to improve the performance of the compression rate. From the experimental results, the proposed method successfully achieves the goal of compression rate improvement.

The rest of this paper is organized as follows. Related work is introduced in Section 2. The proposed method is presented in Section 3. Section 4 summarizes the experimental results and conclusions are drawn in Section 5.

2. Related Work

2.1 Vector Quantization

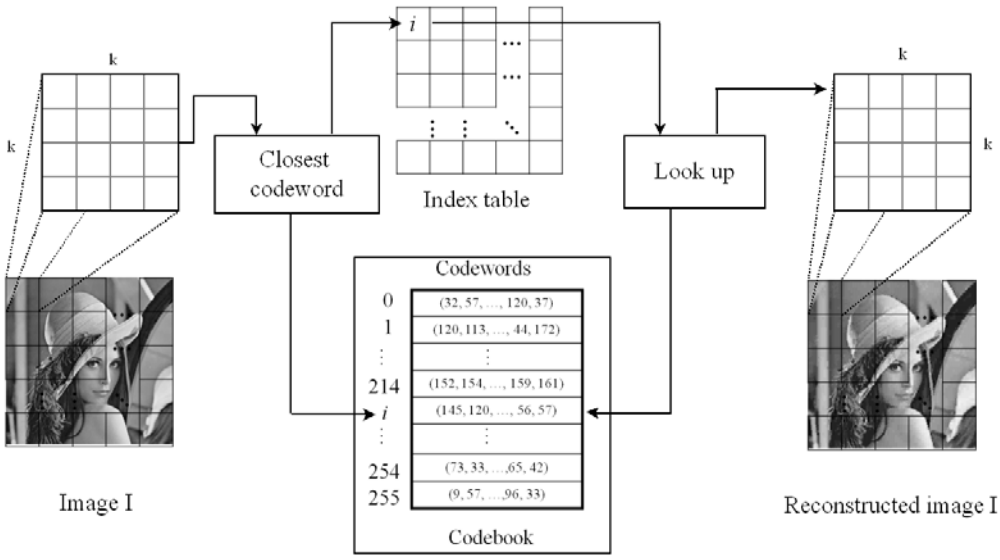


Fig. 1. The concept of VQ compression

Vector quantization (VQ) is a lossy image compression technique. Here, the file size of image can be significantly reduced by encoding index table which generated by VQ compression. Therefore, the storage requirement of an image can be saved. **Fig. 1** shows the VQ encoding and decoding processes. Let a codebook $Y = \{y_i \mid i = 0, 1, \ldots, m\text{-}1\}$ contains m codewords and each codeword contains $k \times k$ dimensions. The key steps of VQ compression are summarized as follows: First, the image I is divided into non-overlapping blocks sized $k \times k$ pixels and denoted as $I = \{B_i \mid I = 0, 1, \ldots, M\text{-}1\}$, where M is the total number of blocks of I. Second, to encode blocks B_i, find a codeword $y_{\min}$ which has the smallest distance between the codeword and B_i. The codeword $y_{\min}$ can be found by using Eq. (1).

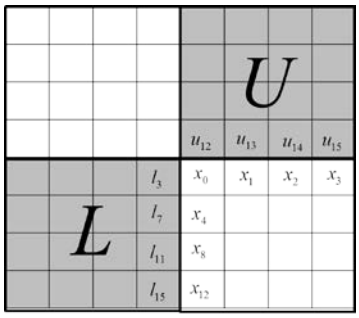
$$y_{\min} = \arg\min_{\forall y_v \in Y} \left\{ \left(\sum_{j=0}^{k\times k-1} (B_i(j) - y_v(j))^2 \right)^{\frac{1}{2}} \right\}, \quad (1)$$

where $y_{\min}$ is the found codeword which is most similar to the encoding block B_i. $B_i(j)$ and $y_v(j)$ represent the j-th pixel B_i and y_v, respectively. Third, B_i is encoded by using the index of $y_{\min}$. After all the blocks have been encoded, the index table is the compression code. The decoding process uses the index in the table to lookup a decoding codeword from the codebook to reconstruct the block. Here, the codebook in the decoding phase is the same as the codebook used in the encoding phase. For instance, if a grayscale image sized 256 × 256 is compressed by VQ with a 256 codewords codebook and a codeword size of 16 pixels, then the compression rate is (total bits of index table) / (total bits of original image) = (((256 × 256) / 16) × 8) / (256 × 256 × 8) = 0.0625.

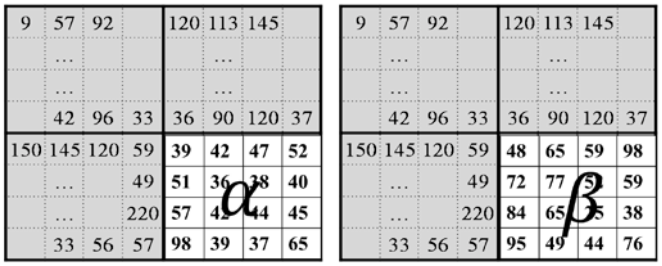
Line and Chang presented a data hiding technique by using the concept of de-clustring [11]. Lin and Chang's method can effectively embed secret data into the compression code and the original VQ index table can be approximately restored when the secret data has been extracted.

2.2 Side-match Vector Quantization (SMVQ)

Side-match vector quantization (SMVQ) is a VQ-based lossy image compression method. The concept of SMVQ is to use the characteristics of a natural image to improve the visual quality of the decompressed image. The characteristic of the natural image means that the pixels of a local area have a similar distribution. **Fig. 2** illustrates the concept of the SMVQ. In **Fig. 2**, the block B_i is the current encoding block and the neighboring border vector (NBV) of block B_i is defined by $NBV = \{x_0 = (u_{12}+l_3)/2, x_1=u_{13}, x_2=u_{14}, x_3=u_{15}, x_4=l_7, *, *, *, x_8=l_{11}, *, *, *, x_{12}=l_{15}, *, *, *\}$ which is obtained from B_i's upper-side (i.e., block U) and left-side (i.e., block L), where the element '*' means; do not care.



(a) The definition of nearboaring block



(b) one pairling block α and β with codeword and 131, respectively.

Fig. 2. The concept of the SMVQ

The key steps of SMVQ are summarized as follows: First, for current encoding block B_i, determine s candidate codewords to form a state codebook by using B_i's NBV. Here, the candidate codeword selection is used to compute the distance between NBV and codeword in Y (i.e., Eq. (2)).

$$dis(NBV_{Bi}, y_v) = \Big(\sum_{j=0,1,2,3,4,8,12} (NBV_{B_i}(j) - y_v(j))^2 \Big)^{\frac{1}{2}}, \tag{2}$$

where v is the index value of codewords in the codebook. Here, the state codebook is denoted as $SCB = \{scw_i \mid i = 0, 1, \ldots, s\text{-}1\}$. Second, a closest codeword of B_i can be found by calculating the distance (i.e., Eq. (3)) between B_i and the codeword in SCB. If the distance between B_i and $scw_{\min}$ is smaller than a predefined threshold, then the block B_i is encoded by SMVQ; otherwise, the block B_i is encoded by VQ.

$$cw_{\min} = \min_{\forall scw_v \in Y} \left\{ \Big(\sum_{j=0}^{k \times k-1} (B_i(j) - scw_v(j))^2 \Big)^{\frac{1}{2}} \right\}. \tag{3}$$

2.3 Locally Adaptive Scheme (LAS)

In 1997, Chang et al. proposed a locally adaptive scheme (LAS) image compression to improve the performance of VQ in terms of compression rate [9]. The main idea of the LAS is to use a history list to play as a codebook cache of codebook and used it to further reduce the size of the compression code. The key steps of the LAS's method are summarized as follows: first, the index table is partitioned into non-overlapping blocks sized $k \times k$ index values. Second, for the current encoding index, check the index value in the history list. If the list contains the same index value, then use the sequence number of index value in the list to encode the index value and move the index value to the front of the list. On the Contrary, if the list has no the index the same with as the current encoding index, then add the current encoding index into the front of the list staying the front. In order to distinguish the confusing indicators with two different cases, one more bit as indicator is needed to put in front of the codeword, to ensure that the correct encoded codes can be retrieved as the same of original one. For instance, if an index can be encoded by using LAS, then the indicator is set as '1'; otherwise, the indicator is set as '0'.

The decoding process is the reverse work of the encoding process. If an indicator taken from compression code equals to '0', then the following $\lceil \log_2 n \rceil$ bits are taken from the compression code to form the VQ codeword index, which is used to reconstruct the decoding block and to add the index into the history list. Here, n is the number of codewords in the codebook. If the indicator is equals to '1', then the following $\lceil \log_2 m \rceil$ bits are taken from the compression code to form the sequence number in index stay in the history list. Note that, m is the size of the history list. According to the sequence number, the codeword index can be found from the history and used it to reconstruct the image block. Also, the corresponding index is moved to the front of the history list.

For example, if a block of VQ index table sized 4×4 is B = {31, 207, 207, 213, 31, 207, 207, 207, 31, 211, 8, 8, 35, 31, 7, 7}, according to the LAS encoding rule, the history index list = {7, 31, 35, 8, 211, 207, 213}. Also, every index value is concatenated with one indicator. So the

compression code is “0 00011111 0 11001111 1 0 0 11010101 1 10 1 10 1 00 1 00 1 01 0 11010011 0 00001000 1 000 0 00100011 1 011 0 00000111 1 000”.

In the LAS decoding rule the indicator is taken from the compression code which is used to indicate the source of the decoding codeword. **Tables 1** and **2** show the entire encoding and decoding process.

Table 1. An example of the LAS encoding process

Step	Input	History index list	Indicator	Output
1	31	31	0	00011111_2
2	207	207, 31	0	11001111_2
3	207	207, 31	1	0_2
4	213	213, 207, 31	0	11010101_2
5	31	31, 213, 207	1	10_2
6	207	207, 31, 213	1	10_2
7	207	207, 31, 213	1	00_2
8	207	207, 31, 213	1	00_2
9	31	31, 207, 213	1	01_2
10	211	211, 31, 207, 213	0	11010011_2
11	8	8, 211, 31, 207, 213	0	00001000_2
12	8	8, 211, 31, 207, 213	1	000_2
13	35	35, 8, 211, 31, 207, 213	0	00100011_2
14	31	31, 35, 8, 211, 207, 213	1	011_2
15	7	7, 31, 35, 8, 211, 207, 213	0	00000111_2
16	7	7, 31, 35, 8, 211, 207, 213	1	000_2

Table 2. An example of the LAS decoding process

Step	Indicator	Input	History index list	Output
1	0	00011111_2	31	31
2	0	11001111_2	207, 31	207
3	1	0_2	207, 31	207
4	0	11010101_2	213, 207, 31	213
5	1	10_2	31, 213, 207	31
6	1	10_2	207, 31, 213	207
7	1	00_2	207, 31, 213	207
8	1	00_2	207, 31, 213	207
9	1	01_2	31, 207, 213	31
10	0	11010011_2	211, 31, 207, 213	211
11	0	00001000_2	8, 211, 31, 207, 213	8
12	1	000_2	8, 211, 31, 207, 213	8
13	0	00100011_2	35, 8, 211, 31, 207, 213	35
14	1	011_2	31, 35, 8, 211, 207, 213	31
15	0	00000111_2	7, 31, 35, 8, 211, 207, 213	7
16	1	000_2	7, 31, 35, 8, 211, 207, 213	7

3. The Proposed Scheme

As mentioned above, the LAS method requires an indicator to indicate the source of a codeword which decreases the benefits of LAS compression. Thus, an indicator elimination

method is proposed to further improve the performance of LAS in terms of the compression rate. The proposed method adopts the concept of a data hiding technique to conceal indicators in the compression code. The proposed method uses a sorted codebook to encode image blocks. Here, any sorting method can be used to sort the codebook. A characteristic of a sorted codebook is that any two adjacent codewords are similar to each other. Before applying the proposed indicator elimination encoding, an image $I = \{p_i \mid i = 1, 2, \ldots, N\}$ sized N pixels is first compressed by VQ, and an index table $T = \{b_i \mid i = 1, 2, \ldots, NB\}$ where NB is the total number of blocks is used.

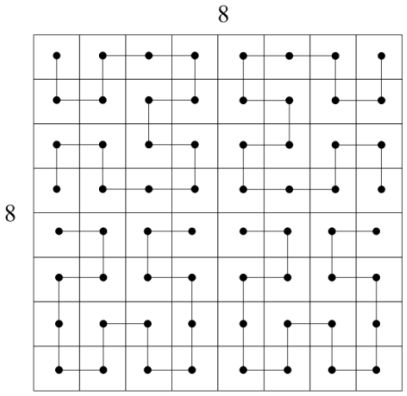


Fig. 3. Fractal-Hilbert-curve scan order with block size 8 × 8

3.1 Scan Order and History List

The traditional LAS method divides the index table into none overlapping blocks which are independent from each other. The LAS method transgresses the characteristic of natural images; that is the local area has a similar pixel distribution. The proposed method accommodates the characteristic to further improve the performance of LAS in terms of compression rate by adopting the Fractal-Hilbert-curve scan order (as shown in **Fig. 3**). Any scan order (e.g., Parallel-scan, Zig-Zag scan, N-scan, Fractal-Hilbert-curve) can be adopted in the proposed method. The reason the proposed method chose the Fractal curve scan order is that the Fractal curve scan order can achieve a better compression rate. Furthermore, the proposed method maintains a history list during the compression and the content of the list is updated by adopting a First-In-First-Out (FIFO) operation.

The history list is sorted by using decreasing order and denoted as $H = \{cw_k \mid k = 0, 1, \ldots, h\text{-}1\}$. Note that, the first and last element in the list (i.e., 0 and N_h-*1*) are reversed for handling special cases. Here, N_h represents the size of the history list. Many history lists maintain strategies can be adopted in the proposed method. The maintain rule of the proposed method is to remove an “oldest” element in the list when a new index is add into a full history list. Here, the oldest means the element that has been in the list the longest time.

3.2 The Encoding Phase

The proposed indicator elimination from LAS encoding is inspired from Chang and Lin’s method [11]. Chang and Lin proposed a data hiding technique to conceal secret data in a VQ index table to achieve the goal of secret data delivery. In the proposed method a side-match-distortion function (SMD) is adopted to conceal the indicators in the compression code, to achieve the goal of indicator elimination.

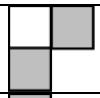
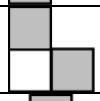
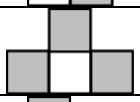
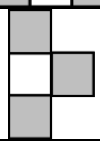
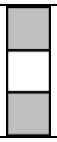
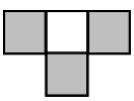
Note that the first block (i.e., b_1) is set as a seed block and encoded by VQ. Before adopting the encoding procedure, the codewords remaining in the VQ codebook and history list are paired. The paring strategy is to sort the codewords in the codebook and to divide the codebook into two parties. The pairing rule is (cw_1, $cw_{N/2}$), (cw_2, $cw_{N/2+1}$), …, ($cw_{N/2-1}$, cw_{N-2}). **Fig. 4** illustrates the example for the codeword pairing. The codewords remaining in the same pair are dissimilar to each other. For simplicity, (α, β) represents a selected codeword pair.

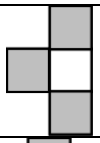
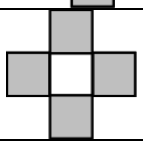
(a) pairing codewords for VQ codebook (b) pairing codewords for history list

Fig. 4. The codewords pairing example

First, the index values are added into the list until the list is full. After that, the current encoding block b_i is either encoded by VQ or LAS. The proposed method adopts the side-match distortion (SMD) function to achieve indicator elimination. Because several scan orders can be used in the encoding phase, the SMD function needs to choose different temporal pixels from b_i. Here Γ is used to denote the set of temporal pixels. According to different scan orders for encoding, block b_i will stay as one of fifteen types (the types are summarized in **Table 3**).

Table 3. The list of total types of current encoding block b_i (i.e., gray block: encoded block; white block: current encoding block)

Types	Pattern	Γ	Types	Pattern	Γ
1		{0, 1, 2, 3}	9		{3, 7, 11, 12, 13, 14, 15}
2		{0, 4, 8, 12}	10		{0, 1, 2, 3, 7, 11, 15}
3		{3, 7, 11, 15}	11		{0, 1, 2, 3, 4, 8, 12}
4		{12, 13, 14, 15}	12		{0, 1, 2, 3, 7, 11, 15}
5		{0, 1, 2, 3, 12, 13, 14, 15}	13		{0, 4, 8, 12, 13, 14, 15}

6		{0, 3, 4, 7, 8, 11, 12, 15}	14		{0, 1, 2, 3, 4, 8, 12}
7		{0, 1, 2, 3, 4, 8, 12}	15		{0, 1, 2, 3, 4, 8, 12}
8		{0, 4, 8, 12, 13, 14, 15}			

The SMD function is defined as follows,

$$\begin{aligned} SMD(\alpha, NBV_{b_i}) &= \sum_{j\in\Gamma} \left|\alpha(j) - NBV_{b_i}(j)\right| \\ SMD(\beta, NBV_{b_i}) &= \sum_{j\in\Gamma} \left|\beta(j) - NBV_{b_i}(j)\right| \end{aligned}, \tag{4}$$

where α and β represent two different blocks reconstructed by the pairing codewords, NBV_{b_i} is the nearboaring block of the current block b_i (see L and U in **Fig. 2** (a)), Γ is the location set of current block b_i, which locatons are closed to the nearboaring blocks.

One example of two blocks with two paring words 4 and 131 are shown in **Fig. 2** (b). The location set Γ are {0, 1, 2, 3, 4, 8, 12}, the pixel value of block α with corresponding location are {39, 42, 47, 52, 51, 57, 98}, and the value of NBV_{b_i} are {(36+59)/2, 90, 120, 37, 51, 57, 98}. Then the $SMD(\alpha, NBV_{b_i})$ can be calculated by the distance measure with α and NBV_{b_i} using (4), 5554 = $|39 - 47.5|^2 + |42 - 90|^2 + |47 - 120|^2 + |52 - 37|^2 + |51 - 51|^2 + |57 - 57|^2 + |98 - 98|^2$. Also, the $SMD(\beta, NBV_{b_i})$ value can be obtained in the same procedure.

According to the source of current encoding block b_i and next block b_{i+1}, the encoding situation can be divided into the following cases:

Case 1: if $IND(b_i) = IND(b_{i+1}) =$ '0' (i.e., the block b_{i+1} cannot be encoded by LAS) and $SMD(\alpha, NBV_{b_i}) \le SMD(\beta, NBV_{b_i})$, then b_i is encoded by using α.

Case 2: if $IND(b_i) = IND(b_{i+1}) =$ '0' and $SMD(\alpha, NBV_{b_i}) > SMD(\beta, NBV_{b_i})$, then b_i is encoded by using 0 || α.

Case 3: if $IND(b_i) =$ '0', $IND(b_{i+1}) =$ '1' (i.e., the block b_{i+1} can be encoded by LAS) and $SMD(\alpha, NBV_{b_i}) \le SMD(\beta, NBV_{b_i})$, then b_i is encoded by using β.

Case 4: if $IND(b_i) =$ '0', $IND(b_{i+1}) =$ '1' and $SMD(\alpha, NBV_{b_i}) > SMD(\beta, NBV_{b_i})$, then b_i is encoded by using m-1 || α.

Case 5: if $IND(b_i) =$ '1', $IND(b_{i+1}) =$ '0' (i.e., the block b_i and b_{i+1} are encoded by LAS and VQ, respectively) and $SMD(\alpha, NBV_{b_i}) \le SMD(\beta, NBV_{b_i})$, then b_i is encoded by using α_h.

Case 6: if $IND(b_i) =$ '1', $IND(b_{i+1}) =$ '0' and $SMD(\alpha, NBV_{b_i}) > SMD(\beta, NBV_{b_i})$, then b_i is encoded by using 0 || α_h.

Case 7: if $IND(b_i) = IND(b_{i+1}) =$ '1' (i.e., the block b_i and b_{i+1} are both encoded by LAS) and $SMD(\alpha, NBV_{b_i}) \le SMD(\beta, NBV_{b_i})$, then b_i is encoded by using β_h.

Case 8: if $IND(b_i) = IND(b_{i+1}) =$ '1' and $SMD(\alpha, NBV_{b_i}) \leq SMD(\beta, NBV_{b_i})$, then b_i is encoded by using $N_{h-1} \parallel \alpha_h$.

Table 4 summarizes all possible cases of the proposed encoding rules.

Table 4. The encoding rules for eight cases

Cases	$IND(b_i)$	$IND(b_{i+1})$	SMD(α, NBV_{b_i}) ≤ SMD(β, NBV_{b_i})	Coding rule
1	0	0	1 (true)	α
2	0	0	0 (false)	$0 \parallel \alpha$
3	0	1	1	β
4	0	1	0	$m\text{-}1 \parallel \alpha$
5	1	0	1	α_h
6	1	0	0	$0 \parallel \alpha_h$
7	1	1	1	β_h
8	1	1	0	$N_h\text{-}1 \parallel \alpha_h$

The key steps of the proposed encoding method are summarized in the following procedure.

Example: LAS Indicator Elimination Encoding

From **Table 5**, the k-th~(k+4)-th VQ index list is L={…, 31, 29, 31, 35, 31, …}, and the corresponding SMD value are {…, 0, 1, 1, 1, 0, …}. Also, the sample of (k-1)-th history list is known and shown in **Fig. 5**. Based on this information, the concatenated compression code is "… 111 010 || 001 || 010 || 00100011 || 00000000 00100010"

Table 5. Known information

	VQ Index Table	SMD(α, NBV_{b_i}) ≤ SMD(β, NBV_{b_i})
k	31	0 (false)
$k+1$	29	1 (true)
$k+2$	31	1
$k+3$	35	1
$k+4$	34	0
$k+5$	39	1

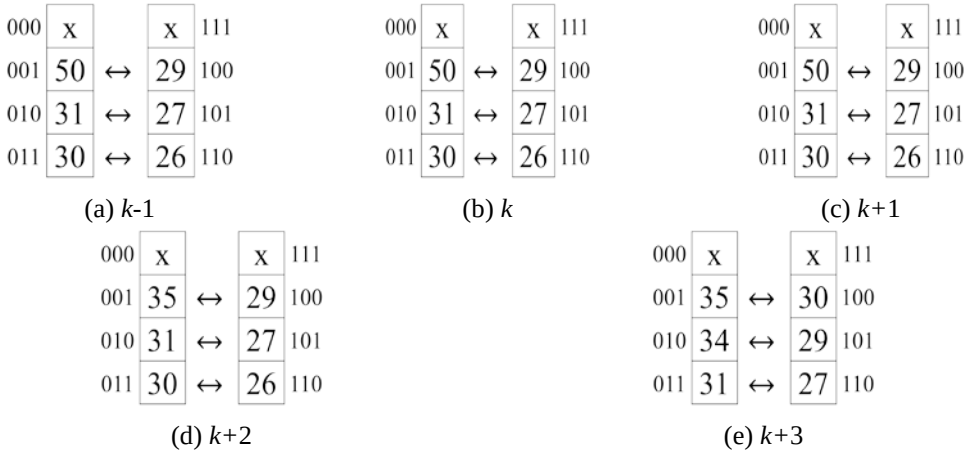


Fig. 5. History list from (k-1)-th to (k+3)-th

3.3 The Decoding Phase

The decoding procedure is the reverse of the encoding procedure. The image can be decompressed by applying the proposed decoding procedure to extract the indicator embedded in the compression code. The key steps of the proposed decoding procedure are described as follows: First, the seed blocks are reconstructed by taking the indices values from the compression code. Before the history list is full, the blocks belong to the seed blocks. In other words, the seed blocks are reconstructed by using the codewords from VQ codebook directory. Second, the remaining blocks are reconstructed using either VQ or LAS. If the decoding index taken from the code stream is equal to 0 or *NB*-1 and $IND(b_i)$ = '0' then the indicator of b_{i+1} is 0 or 1, respectively. In this case the following $\lceil \log_2 N_B \rceil$ bits are used to reconstruct block b_i. Furthermore, if the decoding index taken from the code stream is equal to 0 or N_h-1 and $IND(b_i)$ = '1' then the indicator of b_{i+1} is 0 or 1, respectively. Also, the following $\lceil \log_2 N_h \rceil$ bits are used to take a codeword from the history list and use it to reconstruct block b_i.

For general cases in the remaining blocks, the SMD function is adopted to extract the indicator of block b_{i+1}. For instance, if $SMD(\alpha, NBV_{b_i}) \le SMD(\beta, NBV_{b_i})$ and the index of codeword is α, then the indicator of b_{i+1} is '1'. Contrary, the indicator of b_{i+1} is '0'.

The key steps of the proposed decoding process are summarized as follows.

Procedure: Decoding
Input: Streams of encoded code *EC* and the pre-shared codebook
Output: Index table *T*

Step1: Take $\lceil \log_2 m \rceil$ bits from *EC* to reconstruct the image block and fill the codeword index into *H*, where m is the size of VQ codebook, repeat until *H* is full.
Step 2: The remaining block reconstruction can be done by one of following cases:
Case 1: If $IND(b_i) == 0$ and $b_i == 0$, then take following $\lceil \log_2 m \rceil$ bits from *EC* to replace b_i, use it to reconstruct the image block, and set $IND(b_{i+1}) = 0$.
Case 2: If $IND(b_i) == 0$ and $b_i = m$-1, then take following $\lceil \log_2 m \rceil$ bits from *EC* to replace b_i, use it to reconstruct the image block, and set $IND(b_{i+1}) = 1$.
Case 3: If $IND(b_i) == 0$ and $SMD(\alpha, NBV_{b_i}) \le SMD(\beta, NBV_{b_i})$, then use α to reconstruct the block, and set $IND(b_{i+1}) = 0$.
Case 4: If $IND(b_i) == 0$ and $SMD(\alpha, NBV_{b_i}) > SMD(\beta, NBV_{b_i})$, then use β to reconstruct the block, and set $IND(b_{i+1}) = 1$.
Case 5: If $IND(b_i) == 1$ and $b_i == 0$, then take following $\lceil \log_2 N_h \rceil$ bits from *EC* to replace b_i, use it to reconstruct the image block, and set $IND(b_{i+1}) = 0$.
Case 6: If $IND(b_i) == 1$ and $b_i = N_h$-1, then take following $\lceil \log_2 N_h \rceil$ bits from EC to replace b_i, use it to reconstruct the image block, and set $IND(b_{i+1}) = 1$.
Case 7: If $IND(b_i) == 1$ and $SMD(\alpha, NBV_{b_i}) \le SMD(\beta, NBV_{b_i})$, then use α to reconstruct the block, and set $IND(b_{i+1}) = 0$.
Case 8: If $IND(b_i) == 1$ and $SMD(\alpha, NBV_{b_i}) > SMD(\beta, NBV_{b_i})$, then use β to reconstruct the block, and set $IND(b_{i+1}) = 1$.
Step 3: Repeat Step 2 until all the blocks have been reconstructed.

Example: LAS indicator elimination decoding

Take the fist compression bit pattern "… 111 010…" for example. As the following bit pattern is "111," it means the current index can be found in the history list (see **Fig. 5**), and the next point $IND(b_{i+1})$ is "1." So, the first restored index is 31 as it is located in the $(010)_2$ position of the history list.

4. Experiment Results and Discussion

In order to evaluate the performance of the proposed method, the encoding methods were simulated using MATLAB software on a Pentium IV 2.6 GHz CPU with 2 GB main memory. Nine test images were used in these simulations, and are listed in **Fig. 6**. The size of test images was 512 × 512 pixels. The test images included smooth images (e.g., Jet(F16), Tiffany) and complex content images (e.g., Baboon, Goldhill) for fair testing the performance on a natural image.

(a) Baboon (b) Barbara (c) Goldhill

(d) Jet(F16) (e) Lena (f) Pepper

(g) Sailboat (h) Tiffany (i) Zelda

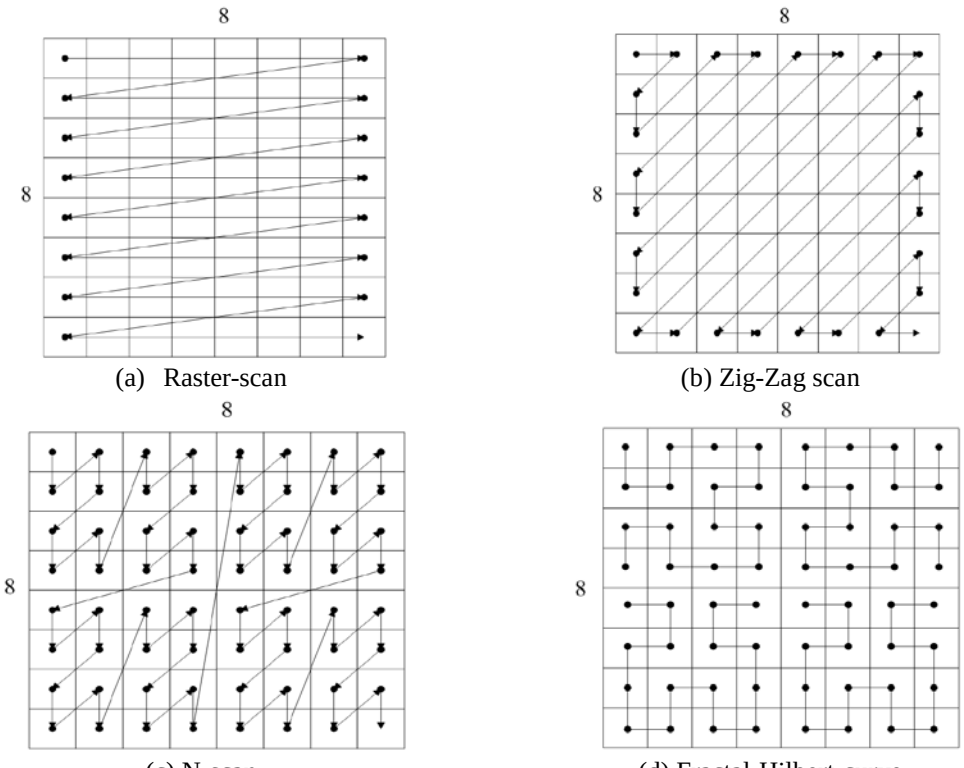
Fig. 6. Nine test images

LAS is adopted to further improve the compression rate of VQ compression. Thus, the visual quality of the LAS compression is the same as VQ compression. Because the purpose of the proposed method is to improve the compression rate of the LAS method, the visual quality of the decompression image generated by the proposed method is the same as the VQ compression method. Furthermore, the compression rate (i.e., denoted as *CR*) is the most important factor for evaluating a compression method. The compression rate is defined as follows:

$$CR = \frac{\|CE\|}{W \times H}, \tag{5}$$

where W and H are the width and height of the image, and $\|CE\|$ represent the total bits of compression code CE.

According to the characteristics of a natural image, the local area has a similar pixel distribution, the local area also has a similar index distribution. Thus, a good scan order can help the index history construction. We implemented Parallel-scan, Zig-Zag scan, N-scan, and Fractal-Hilbert-curve to determine a suitable scan in order to improve compression rate as much as possible. **Fig. 7** illustrates the implemented scan orders. **Table 6** summarizes the number of indices that can be compressed by LAS. Because Fractal-Hilbert-curve scan order tries to visit the neighboring VQ index first, thus it can as many as possible retain recently used indexes in the history list. **Table 6** shows that Fractal-Hilbert-curve is the most suitable for our method, as this scan order can achieve the greatest probability to retain the recently used indexes in history list. **Table 7** summarizes the bit-rate comparison for testing the different scan order. From experimental results, the Fractal-Hilbert-curve can help LAS to achieve a good bit rate. Thus, we suggest that the Fractal-Hilbert-curve scan order is a suitable scan order in the proposed method.



(a) Raster-scan (b) Zig-Zag scan

(c) N-scan (d) Fractal-Hilbert-curve

Fig. 7. Four types of scan order

Table 6. The number of indices compressed by our method (image size 512 × 512)

Images	Parallel-scan	Zig-Zag scan	N-scan	Fractal-Hilbert-curve
Baboon	3341	3113	4201	4291
Barbara	6260	5637	7928	8140
GoldHill	7361	5288	7610	7998
Jet(F16)	10783	9658	11080	11351
Lena	6904	7494	10346	10527
Pepper	7813	7410	10243	10510
Sailboat	8316	7506	9050	9259
Tiffany	10928	10178	12241	12717
Zelda	7671	7366	10135	10433

Table 7. The bit rate testing for different scan orders (image size 512 × 512)

Images	Parallel-scan	Zig-Zag scan	N-scan	Fractal-Hilbert-curve
Baboon	0.4691	0.4725	0.4557	0.4538
Barbara	0.4092	0.4199	0.3804	0.377
GoldHill	0.3746	0.4092	0.3697	0.3643
Jet(F16)	0.3056	0.3267	0.3017	0.2967
Lena	0.3788	0.3679	0.3168	0.3138
Pepper	0.3592	0.3671	0.3165	0.3118
Sailboat	0.3632	0.3764	0.35	0.3467
Tiffany	0.2977	0.3135	0.2777	0.2688
Zelda	0.359	0.3642	0.3147	0.3094

The size of the history list is an important factor in helping LAS compression. **Table 8** and **Table 9** summarize the simulation results for the number of indices encoded by LAS, and the compression rate, respectively. As we see, a large index history list can provide more help for LAS encoding. However, the compression rate may not have a better compression outcome, because a large index history list needs more bits to represent the location number in the history list. Thus, to consider the computation cost and compression rate, we suggest that a suitable size for the index history list is eight.

Table 8. The simulation results for different size of index history list

Images	The Size of History List		
	4	8	16
Baboon	2280	4291	6390
Barbara	6025	8140	9854
GoldHill	5463	7998	10030
Jet(F16)	9642	11351	12439
Lena	8110	10527	11858
Pepper	8145	10510	11862
Sailboat	7389	9259	10532
Tiffany	10377	12717	14124
Zelda	7802	10433	12158

Table 9. The bit rate in different history list size

Images	The Size of History List		
	4	8	16
Baboon	0.4803	0.4538	0.4400
Barbara	0.3959	0.3770	0.3806
GoldHill	0.3894	0.3643	0.3620
Jet(F16)	0.3018	0.2967	0.3213
Lena	0.3346	0.3138	0.3299
Pepper	0.3309	0.3118	0.3266
Sailboat	0.3601	0.3467	0.3587
Tiffany	0.2848	0.2688	0.2917
Zelda	0.3351	0.3094	0.3197

The proposed method fixed the size of the index history list. The index history list maintenance significantly affects the performance of the proposed method in terms of compression rate. Here, we test two maintenance rules: 1) to retire the index with the lowest frequency of use; 2) to retire the index with the longest time in the history list. **Table 10** summarizes the number of indices retired by different maintenance rules. **Table 11** summarizes the compression rate by adopting different index history list maintenance rules. From experimental results, rule 2 is adopted in the proposed improved LAS compression method.

Table 10. The number of index values that can be compressed in the two cases

Images	Rule 1		Rule 2	
	Compressible	Incompressible	Compressible	Incompressible
Baboon	2870	13514	4291	12093
Barbara	5894	10490	8140	8244
GoldHill	5819	10565	7998	8386
Jet(F16)	10300	6084	11351	5033
Lena	7794	8590	10527	5857
Pepper	7837	8547	10510	5874
Sailboat	7354	9030	9259	7125
Tiffany	9022	7362	12717	3667
Zelda	7940	8444	10433	5951

Table 11. The bit rate comparison for different maintenance rules

Images	Rule 1	Rule 2
Baboon	0.4755	0.4538
Barbara	0.4137	0.3770
GoldHill	0.3985	0.3643
Jet(F16)	0.3169	0.2967
Lena	0.3612	0.3138
Pepper	0.3589	0.3118
Sailboat	0.3807	0.3467
Tiffany	0.3326	0.2688
Zelda	0.3533	0.3094

In **Table 12** we can find that the number of index values that can be compressed in the proposed method is higher than in the traditional LAS method in the nine test images. This demonstrates that the proposed method has more index values that can be encoded using fewer bits (3-bit or 6-bit), to achieve a better compression rate.

Table 12. Comparison of the compression rate results for traditional LAS and the proposed method

Images	LAS		Proposed Method	
	Compressible	Incompressible	Compressible	Incompressible
Baboon	4141	12243	4291	12093
Barbara	7424	8960	8140	8244
GoldHill	7274	9110	7998	8386
Jet(F16)	9945	6439	11351	5033
Lena	9223	7161	10527	5857
Pepper	9205	7179	10510	5874
Sailboat	8040	8344	9259	7125
Tiffany	11065	5319	12717	3667
Zelda	9104	7280	10433	5951

In the traditional LAS method, every index needs one indicator, so the indicator rate is 1. The indicator rate is calculated by the number of indicators / the size of index table. **Table 13** summarizes the indicator rate comparison of traditional LAS and the proposed method. The proposed method tries to embed the LAS indicators into compression code to eliminate the indicators by using the data hiding technique. However, the proposed method needs some extra information to handle the special cases. The experimental results show that the number of extra information bits is less than the number of LAS indicator bits.

Table 13. The indicator rate in the traditional LAS method and proposed method

Images	LAS	Proposed Method
Baboon	1	0.808
Barbara	1	0.776
GoldHill	1	0.584
Jet(F16)	1	0.352
Lena	1	0.440
Pepper	1	0.376
Sailboat	1	0.568
Tiffany	1	0.472
Zelda	1	0.304

Table 14 summarizes the comparison of the compression rate of the traditional LAS method and the proposed method. From experimental results, we found that the proposed method work very well in the case of the local area has similar color distribution (i.e., the smooth image content such as Pepper, Tiffany, etc.). In the case of the local area has dissimilar color distribution, the proposed method can't get too much image compression performance improvement (e.g., Baboon). From the experimental results, the proposed method has better compression rate performance than the traditional LAS method.

Table 15 shows the comparison of compression rate *CR* with some recent representive techniques. In VQ method, all indexes are encoded in 8 bits, so the compression bit rate is the same as 0.5.

Chang et al.'s method [21] is based on SMVQ technique which can estimate the indicator from SMVQ, to further improve the compresstion rate. Chang et al.'s method [8] performs better than VQ, SMVQ and LAS. From experimental results it is clear that our method can perform better compression rate than others.

Table 14. Comparison of the bit rate between the LAS method and the proposed method

Images	LAS (A)	Proposed Method (B)	Gain (A-B)
Baboon	0.498	0.454	0.044
Barbara	0.423	0.377	0.046
GoldHill	0.431	0.364	0.067
Jet(F16)	0.348	0.297	0.051
Lena	0.385	0.314	0.071
Pepper	0.383	0.312	0.071
Sailboat	0.402	0.347	0.055
Tiffany	0.337	0.269	0.068
Zelda	0.389	0.309	0.08

Table 15. Comparison of the bit rate between recent techniques and the proposed method

Images	VQ	SMVQ [7]	LAS [2]	Chang et al., 2009 [8]	Chang et al., 2013 [21]	Proposed
Baboon	0.5	0.536	0.498	0.465	0.490	0.454
Barbara	0.5	0.480	0.423	0.440	0.455	0.377
Goldhill	0.5	0.498	0.431	0.446	0.454	0.364
Jet(F16)	0.5	0.401	0.348	0.360	0.364	0.297
Lena	0.5	0.421	0.385	0.417	0.389	0.314
Pepper	0.5	0.422	0.383	0.422	0.395	0.312
Sailboat	0.5	0.459	0.402	0.424	0.443	0.347
Tiffany	0.5	0.401	0.337	0.371	0.354	0.269
Zelda	0.5	0.438	0.389	0.427	0.403	0.309

For the definition of image quality, *PSNR* (peak signal-to-noise ratio) is used to measure the visual quality. The *PSNR* is defined as,

$$PSNR = 10 \times \log_{10} \frac{255^2}{MSE}(dB), \tag{6}$$

where *MSE* (mean-square error) is used to determin the diffirence with two different image I and I with the same height H and width W.

$$MSE = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} (I_{ij} - I'_{ij})^2, \tag{7}$$

where I_{ij} and I'_{ij} represent the two image in location (i, j), respectively. A large *PSNR* value represent the higher visual quality of image.

Table 16 demonstrates that the visual quality of SMVQ, LAS, Chang et al.'s method [8] and our method is the same with VQ, it is meant that our method can retrieve the original index table by proposed encoding/decoding phase.

5. Conclusion

In this paper we proposed a novel method to eliminate the indicators of LAS. Although LAS

has good performance, the extra indicators of LAS still affect the compression rate. To improve the compression rate of LAS, the proposed method uses the concept of a data hiding method to eliminate indicators. From the experimental results, the proposed method successfully improves the performance of the compression rate by eliminating the indicators of LAS. Moreover, compared with traditional LAS, our proposed method is more flexible and improves the compression rate.

Table 16. Comparison of the *PSNR* value (dB) between between recent techniques and the proposed method

Images	VQ	SMVQ [7]	LAS [2]	Chang et al., 2009 [8]	Chang et al., 2013[21]	Proposed
Baboon	25.46	25.46	25.46	25.46	25.46	25.46
Barbara	24.93	24.92	24.93	24.93	24.92	24.93
Goldhill	28.91	28.91	28.91	28.91	28.91	28.91
Jet(F16)	27.13	27.13	27.13	27.13	27.13	27.13
Lena	28.65	28.65	28.65	28.65	28.65	28.65
Pepper	28.45	28.45	28.45	28.45	28.00	28.45
Sailboat	25.01	25.01	25.01	25.01	25.00	25.01
Tiffany	30.24	30.24	30.24	30.24	30.23	30.24
Zelda	31.31	31.31	31.31	31.31	31.31	31.31

References

[1] C. H. Hsieh, J. C. Tsai, "Lossless compression of VQ index with search-order coding," *IEEE Transactions on Image Professing*, vol. 5, pp. 1579-1582, November 1996. Article (CrossRef Link).

[2] J. L. Bentley, D. D. Sleator, R. E. Tarjan, V. K. Wei, "A locally adaptive data compression scheme," *Commun. ACM*, vol. 29, pp. 320-330, April 1986. Article (CrossRef Link).

[3] S. S. Maniccam, N. Bourbakis, "Lossless compression and information hiding in images," *Pattern Recognition*, vol. 37, pp. 475-486, March 2001. Article (CrossRef Link).

[4] G.K. Wallace, "JPEG still image data compressing standard," *IEEE Transactions on Consumer Electronics*, vol. 38, Feburary 1992. Article (CrossRef Link).

[5] R. Gary, "Vector quantization," *IEEE ASSP Magazine*, vol. 1, pp. 4-29, April 1984. Article (CrossRef Link).

[6] Y. Linde, "An algorithm for vector quantize design," *IEEE Transactions on Communication*, vol. 28, pp. 84-95, 1980. Article (CrossRef Link).

[7] T. Kim, "Side match and overlap match vector quantizes for images," *IEEE Transactions on Image Professing*, vol. 1, pp. 170-185, April 1992. Article (CrossRef Link).

[8] C. C. Chang, Y. C. Chou, Y. P. Hsieh, "Search-order coding method with indicator-elimination property," *Journal of Systems and Software*, vol. 82, pp. 516-525, March 2009. Article (CrossRef Link).

[9] C. C. Chang, C. H. Sung, T. S. Chen, "A locally adaptive scheme for image index compression," in *Proc. of the 1997 Conference on Computer Vision, Graphics, and Image Processing*, pp. 93-99, August 1997. Article (CrossRef Link).

[10] H. S. Tseng, C. C. Chang, "Error resilient locally adaptive data compression," *Journal of Systems and Software*, no. 8, pp. 1156-1160, August 2006. Article (CrossRef Link).

[11] C. Y. Lin, C. C. Chang, "Hiding data in VQ-compressed images using dissimilar pairs," *Journal of computers*, vol. 17, pp. 3-10, June 2008. Article (CrossRef Link).

[12] C. C. a. L. Chen, "Hiding data in images by simple LSB substitution," *Pattern recognition*, vol. 37, pp. 469-474, March 2001. Article (CrossRef Link).

[13] C. C. Lin, W. H. Tsai, "Secret image sharing with steganography and authentication," *Journal of Systems and Software*, vol. 73, pp. 405-414, November 2004. Article (CrossRef Link).
[14] C. C. Thein, J. C. Lin, "A simple and high-hiding capacity method for hiding digit-by-digit data in images based on modulus function," *Pattern Recognition*, vol. 36, pp. 2875-2881, December 2003. Article (CrossRef Link).
[15] R. Z. Wang, C. F. Lin, G. C. Lin, "Image hiding by optimal LSB substitution and genetic algorithm," *Pattern Recognition*, vol. 34, pp. 671-683, March 2001. Article (CrossRef Link).
[16] R. Z. Wang, C. F. Lin, G. C. Lin, "Hiding data in images by optimal moderately-significant-bit replacement," *IEE Electronics Letters*, vol. 36, pp. 2069-2070, December 2000. Article (CrossRef Link).
[17] D. C. Wu, W. H. Tsai, "Spatial-domain image hiding using image differencing," *IEE Proceeding on Vision, Image and Signal Processing*, vol. 147, pp. 29-37, February 2000. Article (CrossRef Link).
[18] Y. S. Wu, C. C. Thein, J. C. Lin, "Sharing and hiding secret images with size constraint," Pattern Precognition, vol. 37, pp. 1377-1385, July 2004. Article (CrossRef Link).
[19] I. C. Lin, Y. B. Lin, C. M. Wang, "Hiding data in spatial domain images with distortion tolerance," *Computer Standards and Interfaces*, vol. 31, pp. 458-464, February 2009. Article (CrossRef Link).
[20] H. W. Tseng, C. P. Hsieh, "Prediction-based reversible data hiding," *Information Sciences*, vol. 79, pp. 2460-2469, June 2009. Article (CrossRef Link).
[21] C. C. Chang, Y. C. Chou, C. Y. Lin, "An indicator elimination method for side-match vector quantization," *Journal of Information Hiding and Multimedia Signal Processing*, Vol. 4, No. 4, Oct. 2013, pp. 233--249. Article (CrossRef Link).

Hon-Hang Chang is Ph. D. student at the Department of Computer Science and Information Engineering, National Central University (NCU), Taiwan (R. O. C.). He acquired his Master's degree in Department of Photonics and Communication Engineering of Asia University of Taiwan in 2011. His research fields are image processing, information hiding and water marking.

Yung-Chen Chou is an Assistant Professor at the Department of Computer Science and Information Engineering of Asia University, Taiwan (R. O. C.). He acquired his Ph. D. degree from National Chung Chen University, Taiwan. His research interests are in the area of Information hiding, watermarking and image processing.

Timothy K. Shih is a Professor at Department of Computer Science and Information Engineering, National Central University, Taiwan. Prof. Shih is a Fellow of the Institution of Engineering and Technology (IET). In addition, he is a senior member of ACM and a senior member of IEEE. Prof. Shih also joined the Educational Activities Board of the Computer Society. He acquired his M. S. in Computer Engineering from California State University, Chico, USA in 1985 and Ph. D. in Computer Engineering from Santa Clara University, USA in 1993. His current research interests are mainly in the area of Computer Vision, Interactive Multimedia, Multimedia Processing, Human Computer Interaction and e-learning.