Journal of Information Hiding and Multimedia Signal Processing ©2019 ISSN 2073-4212
Ubiquitous International Volume **10**, Number **1**, January **2019**

A Novel Image Compression Method Using Turtle-Shell-Shaped Codewords with Less Blocking Artifact

Ching-Chun Chang
Department of Computer Science
University of Warwick, Coventry CV4 7AL, UK
C.Chang.2@warwick.ac.uk

Yanjun Liu
Department of Information Engineering and Computer Science
Feng Chia University, Taichung 407, Taiwan, R.O.C.
yjliu104@gmail.com

Xiao-Qian Yang
Department of Computer Science and Information Engineering
National Chung Cheng University, Chiayi 62102, Taiwan, R.O.C.
tiffany61706@yahoo.com.tw

Received September 2017; revised August 2018

Abstract. *Compression techniques can encode digital data and diminish the quantity, thus they have undergone significant development in the past ten years. Vector quantization (VQ) is a well-known image compression method that encodes each non-overlapping image block by the VQ indices. In the decompression phase, the VQ index is mapped to the corresponding codeword to restructure a similar image block. However, there is a block artifact in the border between adjacent reconstructed blocks. To solve this problem, we proposed a turtle-shell-shaped division method that divides the image into several overlapping blocks. Each block can be compressed as one VQ index. Different from the traditional VQ compression method, the border between adjacent image blocks is recovered by the mean of the codewords. Therefore, there is no visual difference in the border between adjacent reconstructed image blocks. Experimental results show that the visual quality of the reconstructed image is better than that of the traditional VQ compression method.*

Keywords: Compression techniques, Vector quantization, Block artifact, Turtle-shell-shaped, Visual quality

1. Introduction.

Compression techniques can be used to encode digital data, e.g., images [1, 2, 3], audio files [4, 5], videos [6, 7], alphabetic characters, and numbers [8] (Fig. 1). In this paper, image compression methods are investigated, and they can be classified into two types, i.e., the frequency domain [9, 10, 11, 12] and the spatial domain [1, 2, 3].

In the frequency domain, the typical compression methods are the discrete Fourier transform (DFT) [9], the discrete cosine transform (DCT) [10], and the discrete wavelet transform (DWT) [11]. In these methods, all of the pixels in the original image are transformed into coefficients. The coefficients in low frequency can effectively represent the image. On the other hand, the coefficients in high frequency only represent the details of the original image. Therefore, the kind of method does not need to record the

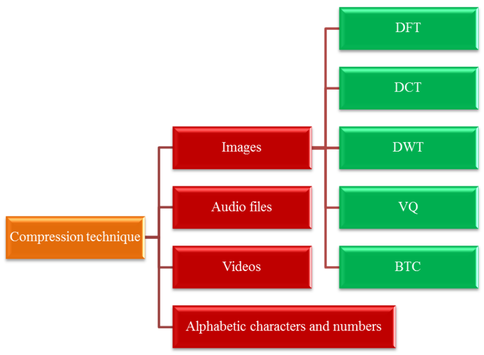


Figure 1. Classification of compression technique

coefficients in high frequency. However, this method involves many multiplication and division operations, which result in high computational costs.

Conversely, in the spatial domain, the compression method does not have complex computational operations. The typical compression methods are block truncation coding (BTC) [1] and vector quantization (VQ) [2]. In the BTC method, each pixel can be compressed as one bit. The compression method is as follows. One image is divided into several blocks, and the average value of the pixels in the block is calculated. Pixels that are equal to or smaller than the average value are replaced by 0. Conversely, pixels that are greater than the average value are replaced by 1.

Linde *et al.* [2] proposed a VQ-based compression method to further reduce the length of the compression code. In their method, the compressed images are divided into several non-overlapping blocks, which are further clustered according to their similarity. Afterwards, the centroids of the clusters are calculated and used as the codeword. In the encoding phase, the image block is mapped into the most similar codeword, and the image block is replaced by the index of the codeword. Therefore, the method can achieve high compression efficiency. However, the visual appearance is significantly different in the borders between the neighboring, reconstructed image blocks. In other words, the visual quality of the VQ method can be improved further.

Thus, we proposed a novel turtle-shell-shaped division method to segment images into several overlapping blocks. Each compressed block is replaced by the index of the most similar codeword. Different from the traditional VQ compression method, the borders between neighboring blocks are reconstructed by means of the codewords, which avoids the block artifact. In other words, the visual quality of the proposed method is much better than that of the VQ method.

2. **VQ Compression Method.**

Fig. 2 shows the three procedures in the VQ compression method, i.e., generation of the codebook, encoding of the image, and reconstruction of the image. The details of these three procedures are presented below.

Generation of the Codebook
Encoding of the Image
Reconstruction of the Image

Figure 2. Flowchart of the VQ compression method

2.1. **Generation of the Codebook.**

The compressed image with the size of $H \times W$ was divided into $\left\lfloor \frac{H}{h} \right\rfloor \times \left\lfloor \frac{W}{w} \right\rfloor$ non-overlapping blocks, each of which consisted of $h \times w$

pixels, i.e., $B = \{P_1, P_2, \ldots, P_{h\times w}\}$. Then, K image blocks were chosen randomly from Bs to be the candidate blocks, and they were used to classify all of the image blocks into K clusters. The classification process was as follows. Let the candidate block be $C_i = \{c_{(i,1)}, c_{(i,2)}, \ldots, c_{(i,h\times w)}\}$, where i denotes the ID number of the candidate block and $1 \le i \le K$. The Euclidean distance, d_i, between the image block B and each candidate block C_i was calculated, i.e., $d_i = \sqrt{\sum_{j=1}^{h\times w}(P_j - c_{(i,j)})^2}$. The block B was merged with the candidate block with the shortest Euclidean distance to be a cluster. After all of the image blocks were classified, the centroid of each cluster was calculated to obtain new candidate blocks, and the classification procedure was repeated until the centroid of each cluster was constant. Let the final centroid, i.e., codeword, be $C'_i = \{c'_{(i,1)}, c'_{(i,2)}, \ldots, c'_{(i,h\times w)}\}$.

2.2. **Encoding of the Image.** In the encoding procedure, the Euclidean distance d_i between the compressed block and each codeword was calculated by the equation $d_i = \sqrt{\sum_{j=1}^{h\times w}(P_j - c'_{(i,j)})^2}$. The index of the shortest Euclidean distance, I, was used as the compression code, because the codeword with the shortest Euclidean distance was very similar to the compressed block.

Fig. 3 shows an example to illustrate the encoding procedure. Figs. 3(a) and 3(b) show a compressed image and a codebook, respectively. The Euclidean distances d_i between the first compressed block and seven codewords were 137.94, 18.55, 19.21, 38.17, 151.84, 5.66, and 182. The sixth Euclidean distance was shorter than the other Euclidean distances, so the compression code was 6. The encoding procedure of the other compressed blocks was the same as that of the first compressed block. Fig. 3(c) shows the VQ index table.

155	155	155	158	152	152	152	153
155	155	155	158	152	152	152	153
155	155	155	158	152	152	152	153
155	155	155	158	152	152	152	153
157	160	159	161	163	162	164	167
157	160	159	161	163	162	164	167
157	160	159	161	163	162	164	167
157	160	159	161	163	162	164	167

(a) Compressed image

Index	Codeword
1	{105, 115, 133, 149, 167, 187, 195, 188, 172, 165, 152, 137, 111, 103, 104, 117}
2	{151, 151, 151, 150, 152, 151, 151, 151, 152, 151, 151, 151, 152, 152, 152, 151}
3	{160, 160, 160, 160, 160, 160, 160, 160, 161, 161, 161, 160, 161, 161, 161, 161}
4	{163, 163, 163, 162, 165, 165, 164, 164, 167, 167, 166, 165, 168, 168, 167, 166}
5	{136, 137, 133, 129, 135, 132, 124, 117, 125, 119, 111, 105, 113, 107, 102, 99}
6	{155, 156, 156, 157, 155, 156, 156, 157, 156, 156, 156, 156, 156, 156, 156, 156}
7	{84, 110, 156, 147, 106, 83, 112, 153, 145, 114, 82, 106, 131, 144, 116, 92}

(b) Codebook

6	2
3	4

(c) VQ index table

FIGURE 3. Example of the encoding procedure of the VQ compression method

2.3. **Reconstruction of the Image.** After obtaining the VQ index table, each index was mapped into the corresponding codeword of the codebook to reconstruct the similar image block. After reconstructing each block, the reconstruction procedure was completed.

As mentioned above, the VQ index table was {6, 2, 3, 4}. Since the first VQ index was 6, the sixth codeword in the codebook, {155, 156, 156, 157, 155, 156, 156, 157, 156, 156, 156, 156, 156, 156, 156, 156}, was used as the first reconstructed block, as shown in Fig. 4. The second index was 2, and the second codeword, {151, 151, 151, 150, 152, 151, 151, 151, 152, 151, 151, 151, 152, 152, 152, 151}, was used to reconstruct the second block. The reconstruction procedure of the other blocks was the same as the above procedure. Fig. 4 shows the reconstructed image.

155	155	156	156	151	152	152	152
156	156	156	156	151	151	151	152
156	156	156	156	151	151	151	152
157	157	156	156	150	151	151	151
160	160	161	161	163	165	167	168
160	160	161	161	163	165	167	168
160	160	161	161	163	164	166	167
160	160	160	161	162	164	165	166

FIGURE 4. Reconstructed image

3. **Proposed Method.** Although the VQ compression method is simple, the visual appearance is significantly different in the border between adjacent reconstructed blocks, and this is called the block artifact. In order to solve this problem, we proposed a turtle-shell-shaped division method, which is inspired from [13]. The proposed method is presented below.

3.1. **Turtle-Shell-Shaped Division Procedure.** The N compressed images sized $H \times W$ is divided into $N \times \lfloor \frac{H-1}{2} \rfloor \times \lfloor \frac{W-1}{2} \rfloor$ overlapping blocks, where $N \geq 1$. Each block consists of eight pixels, i.e., $B_{(x,y)} = \{P_1, P_2, \ldots, P_8\}$, and (x, y) denotes the ID number of the divided block. The shape of the divided blocks is similar to a turtle shell, as shown in Fig. 5. Note that a few pixels in the border of the image were not divided, and they are not compressed and recorded. In the decompression phase, the incompressible pixels will be recovered by the neighboring pixels.

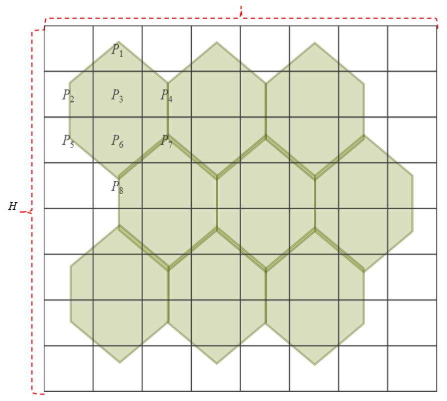
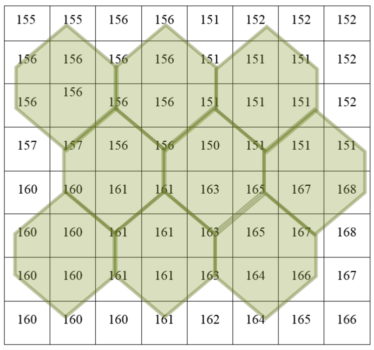


FIGURE 5. Structure of the divided block

According to the random seed, K image blocks are chosen from $B_{(x,y)}$ to be the candidate blocks, and they are used to classify all of the blocks into K clusters. The classification process is as follows. Let the candidate block be $C_i = \{c_{(i,1)}, c_{(i,2)}, \ldots, c_{(i,8)}\}$, where i denotes the ID number of the candidate block and $1 \leq i \leq K$. The Euclidean distance d_i between the pixels in the block $B(x, y)$ and each candidate block C_i is calculated, i.e., $d_i = \sqrt{\sum_{j=1}^{8} (P_j - c_{(i,j)})^2}$. The block $B_{(x,y)}$ is merged with the candidate block with the shortest Euclidean distance to be a cluster. After all blocks are classified, the centroid of each cluster is calculated to obtain new candidate blocks, and the classification procedure is repeated until the the current classification result is the same as the previous classification result. Let the final centroid, i.e., codeword, be $C'_i = \{c'_{(i,1)}, c'_{(i,2)}, \ldots, c'_{(i,8)}\}$.

3.2. **Compression Procedure.** After generating the K codewords, the Euclidean distance, d_i, between the pixels in the compressed block $B_{(x,y)}$ and each codeword C'_i is calculated, i.e., $d_i = \sqrt{\sum_{j=1}^{8}(P_j - c'_{(i,j)})^2}$. The shorter Euclidean distance implies that the degree of similarity between the compressed block and the codeword is very high. Therefore, the index of the codeword with the shortest Euclidean distance, I, is used as the compression code of the block $B_{(i,j)}$.

Fig. 6 shows an example of the encoding procedure. The set of eight pixels in the first block are {155, 156, 156, 156, 156, 156, 156, 157}, thus the Euclidean distances between the pixels in the first block and each codeword are 14.97, 3.16, 29.78, 22, 8.6, 153.45, and 141.23. Since the second Euclidean distance is shorter than the other Euclidean distances, the index "2" of the second Euclidean distance is used as the compression code. The encoding procedure of the other blocks is the same as that of the first block. Fig. 6(c) shows the VQ index table.



155	155	156	156	151	152	152	152
156	156	156	156	151	151	151	152
156	156	156	156	151	151	151	152
157	157	156	156	150	151	151	151
160	160	161	161	163	165	167	168
160	160	161	161	163	165	167	168
160	160	161	161	163	164	166	167
160	160	160	161	162	164	165	166

(a) An original image

Index	Codeword
1	{151, 150, 151, 151, 150, 151, 151, 151}
2	{155, 155, 155, 155, 155, 155, 155, 155}
3	{143, 162, 155, 149, 172, 169, 162, 171}
4	{164, 164, 164, 163, 164, 164, 163, 164}
5	{159, 159, 159, 159, 159, 159, 159, 159}
6	{102, 101, 101, 102, 102, 102, 102, 102}
7	{83, 156, 162, 155, 111, 105, 104, 71}

(b) Codebook

2	2	1
5	5	3
5	4	4

(c) VQ index table

Figure 6. Example of the encoding procedure of the proposed method

3.3. **Decompression Procedure.** In the decompression procedure, each VQ index I can be mapped into the corresponding codeword $\{c'_{(I,1)}, c'_{(I,2)}, \ldots, c'_{(I,8)}\}$, thereby reconstructing the block $B_{(x,y)}$. Note that the overlapping region between adjacent blocks is reconstructed by means of several codewords, rather than only one codeword. Therefore, there is no visual difference in the borders between the reconstructed blocks. The details of the reconstruction procedure are presented below.

Fig. 7 shows the current VQ index I, the eight neighboring VQ indices and their codewords. For example, the the top-left VQ index is TL and its codeword is $\{tl_1, tl_2, \ldots, tl_8\}$. The procedure of reconstructing the blocks is presented below.

$TL = \{tl_1, tl_2, ..., tl_8\}$	$T = \{t_1, t_2, ..., t_8\}$	$TR = \{tr_1, tr_2, ..., tr_8\}$
$L = \{l_1, l_2, ..., l_8\}$	$I = \{c'_{(I,1)}, c'_{(I,2)}, ..., c'_{(I,8)}\}$	$R = \{r_1, r_2, ..., r_8\}$
$BL = \{bl_1, bl_2, ..., bl_8\}$	$B = \{b_1, b_2, ..., b_8\}$	$BR = \{br_1, br_2, ..., br_8\}$

Figure 7. Current VQ index I and its neighboring VQ indices

Case 1: : If the ID number of the reconstructed block, (x, y), is equal to $(1, 1)$, the codewords of the three VQ indices I, R, and B are used to reconstruct the block. The equations used to reconstruct the eight pixels are $P_1 = c'_{(I,1)}, P_2 = c'_{(I,2)}, P_3 = c'_{(I,3)}, P_4 = \left\lfloor \frac{c'_{(I,4)}+r_2}{2} \right\rfloor, P_5 = c'_{(I,5)}, P_6 = c'_{(I,6)}, P_7 = \left\lfloor \frac{c'_{(I,7)}+r_5+b_1}{3} \right\rfloor$ and $P_8 = \left\lfloor \frac{c'_{(I,8)}+b_2}{2} \right\rfloor$.

Case 2: : If $x = 1$ and $1 < y < \lfloor\frac{W-1}{2}\rfloor$, the codewords of the five VQ indices I, L, R, BL, and B are used to reconstruct the block. The equations used to reconstruct the eight pixels are $P_1 = c'_{(I,1)}, P_2 = \left\lfloor\frac{c'_{(I,2)}+l_4}{2}\right\rfloor, P_3 = c'_{(I,3)}, P_4 = \left\lfloor\frac{c'_{(I,4)}+r_2}{2}\right\rfloor, P_5 = \left\lfloor\frac{c'_{(I,5)}+l_7+bl_1}{3}\right\rfloor, P_6 = c'_{(I,6)}, P_7 = \left\lfloor\frac{c'_{(I,7)}+r_5+b_1}{3}\right\rfloor$ and $P_8 = \left\lfloor\frac{c'_{(I,8)}+b_2+bl_4}{3}\right\rfloor$.

Case 3: : If $(x, y) = (1, \lfloor\frac{W-1}{2}\rfloor)$, the codewords of the four VQ indices I, L, BL, and B are used to reconstruct the block. The equations used to reconstruct the eight pixels are $P_1 = c'_{(I,1)}, P_2 = \left\lfloor\frac{c'_{(I,2)}+l_4}{2}\right\rfloor, P_3 = c'_{(I,3)}, P_4 = c'_{(I,4)}, P_5 = \left\lfloor\frac{c'_{(I,5)}+l_7+bl_1}{3}\right\rfloor, P_6 = c'_{(I,6)}, P_7 = \left\lfloor\frac{c'_{(I,7)}+b_1}{2}\right\rfloor$ and $P_8 = \left\lfloor\frac{c'_{(I,8)}+b_2+bl_4}{3}\right\rfloor$.

Case 4: : If $1 < x < \lfloor\frac{H-1}{2}\rfloor, y = 1$, and x is an odd number, the codewords of the four VQ indices I, T, R, and B are used to reconstruct the block. The equations used to reconstruct the eight pixels are $P_1 = \left\lfloor\frac{c'_{(I,1)}+t_5}{2}\right\rfloor, P_2 = c'_{(I,2)}, P_3 = c'_{(I,3)}, P_4 = \left\lfloor\frac{c'_{(I,4)}+r_2+t_8}{3}\right\rfloor, P_5 = c'_{(I,5)}, P_6 = c'_{(I,6)}, P_7 = \left\lfloor\frac{c'_{(I,7)}+b_1+r_5}{3}\right\rfloor$ and $P_8 = \left\lfloor\frac{c'_{(I,8)}+b_2}{2}\right\rfloor$.

Case 5: : If $1 < x < \lfloor\frac{H-1}{2}\rfloor, 1 < y < \lfloor\frac{W-1}{2}\rfloor$, and x is an odd number, the codewords of the seven VQ indices I, TL, T, L, R, BL, and B are used to reconstruct the block. The equations used to reconstruct the eight pixels are $P_1 = \left\lfloor\frac{c'_{(I,1)}+tl_7+t_5}{3}\right\rfloor, P_2 = \left\lfloor\frac{c'_{(I,2)}+tl_8+l_4}{3}\right\rfloor, P_3 = c'_{(I,3)}, P_4 = \left\lfloor\frac{c'_{(I,4)}+r_2+t_8}{3}\right\rfloor, P_5 = \left\lfloor\frac{c'_{(I,5)}+l_7+bl_1}{3}\right\rfloor, P_6 = c'_{(I,6)}, P_7 = \left\lfloor\frac{c'_{(I,7)}+b_1+r_5}{3}\right\rfloor$ and $P_8 = \left\lfloor\frac{c'_{(I,8)}+b_2+bl_4}{3}\right\rfloor$.

Case 6: : If $1 < x < \lfloor\frac{H-1}{2}\rfloor$ and $y = \lfloor\frac{W-1}{2}\rfloor$, and x is an odd number, the codewords of the six VQ indices I, TL, T, L, BL, and B are used to reconstruct the block. The equations used to reconstruct the eight pixels are $P_1 = \left\lfloor\frac{c'_{(I,1)}+tl_7+t_5}{3}\right\rfloor, P_2 = \left\lfloor\frac{c'_{(I,2)}+tl_8+l_4}{3}\right\rfloor, P_3 = c'_{(I,3)}, P_4 = \left\lfloor\frac{c'_{(I,4)}+t_8}{2}\right\rfloor, P_5 = \left\lfloor\frac{c'_{(I,5)}+l_7+bl_1}{3}\right\rfloor, P_6 = c'_{(I,6)}, P_7 = \left\lfloor\frac{c'_{(I,7)}+b_1}{2}\right\rfloor$ and $P_8 = \left\lfloor\frac{c'_{(I,8)}+b_2+bl_4}{3}\right\rfloor$.

Case 7: : If $x = \lfloor\frac{H-1}{2}\rfloor$ and $y = 1$, the codewords of the three VQ indices I, T, and R are used to reconstruct the block. The equations used to reconstruct the eight pixels are $P_1 = \left\lfloor\frac{c'_{(I,1)}+t_5}{2}\right\rfloor, P_2 = c'_{(I,2)}, P_3 = c'_{(I,3)}, P_4 = \left\lfloor\frac{c'_{(I,4)}+t_8+r_2}{3}\right\rfloor, P_5 = c'_{(I,5)}, P_6 = c'_{(I,6)}, P_7 = \left\lfloor\frac{c'_{(I,7)}+r_5}{2}\right\rfloor$ and $P_8 = c'_{(I,8)}$.

Case 8: : If $x = \lfloor\frac{H-1}{2}\rfloor$ and $1 < y < \lfloor\frac{W-1}{2}\rfloor$, the codewords of the five VQ indices I, TL, T, L, and R are used to reconstruct the block. The equations used to reconstruct the eight pixels are $P_1 = \left\lfloor\frac{c'_{(I,1)}+t_5+tl_7}{3}\right\rfloor, P_2 = \left\lfloor\frac{c'_{(I,2)}+tl_8+l_4}{3}\right\rfloor, P_3 = c'_{(I,3)}, P_4 = \left\lfloor\frac{c'_{(I,4)}+t_8+r_2}{3}\right\rfloor, P_5 = \left\lfloor\frac{c'_{(I,5)}+l_7}{2}\right\rfloor, P_6 = c'_{(I,6)}, P_7 = \left\lfloor\frac{c'_{(I,7)}+r_5}{2}\right\rfloor$ and $P_8 = c'_{(I,8)}$.

Case 9: : If $(x, y) = (\lfloor\frac{H-1}{2}\rfloor, \lfloor\frac{W-1}{2}\rfloor)$, the codewords of the four VQ indices I, TL, T, and L are used to reconstruct the block. The equations used to reconstruct the eight pixels are $P_1 = \left\lfloor\frac{c'_{(I,1)}+t_5+tl_7}{3}\right\rfloor, P_2 = \left\lfloor\frac{c'_{(I,2)}+tl_8+l_4}{3}\right\rfloor, P_3 = c'_{(I,3)}, P_4 = \left\lfloor\frac{c'_{(I,4)}+t_8}{2}\right\rfloor, P_5 = \left\lfloor\frac{c'_{(I,5)}+l_7}{2}\right\rfloor, P_6 = c'_{(I,6)}, P_7 = c'_{(I,7)}$ and $P_8 = c'_{(I,8)}$.

Case 10: : If $y = 1$, and x is an even number, the codewords of the six VQ indices I, TR, T, R, B, and BR are used to reconstruct the block. The equations used to reconstruct the eight pixels are $P_1 = \left\lfloor\frac{c'_{(I,1)}+t_7+tr_5}{3}\right\rfloor, P_2 = \left\lfloor\frac{c'_{(I,2)}+t_8}{2}\right\rfloor, P_3 =$

$c'_{(I,3)}, P_4 = \left\lfloor \frac{c'_{(I,4)}+tr_8+r_2}{3} \right\rfloor, P_5 = \left\lfloor \frac{c'_{(I,5)}+b_1}{2} \right\rfloor, P_6 = c'_{(I,6)}, P_7 = \left\lfloor \frac{c'_{(I,7)}+r_5+br_1}{3} \right\rfloor$ and $P_8 = \left\lfloor \frac{c'_{(I,8)}+b_4+br_2}{3} \right\rfloor$.

Case 11: : If $1 < y < \lfloor \frac{W-1}{2} \rfloor$ and x is an even number, the codewords of the seven VQ indices I, TR, T, L, R, B, and BR are used to reconstruct the block. The equations used to reconstruct the eight pixels are $P_1 = \left\lfloor \frac{c'_{(I,1)}+t_7+tr_5}{3} \right\rfloor, P_2 = \left\lfloor \frac{c'_{(I,2)}+t_8+l_4}{3} \right\rfloor, P_3 = c'_{(I,3)}, P_4 = \left\lfloor \frac{c'_{(I,4)}+tr_8+r_2}{3} \right\rfloor, P_5 = \left\lfloor \frac{c'_{(I,5)}+b_1+l_7}{3} \right\rfloor, P_6 = c'_{(I,6)}, P_7 = \left\lfloor \frac{c'_{(I,7)}+r_5+br_1}{3} \right\rfloor$ and $P_8 = \left\lfloor \frac{c'_{(I,8)}+b_4+br_2}{3} \right\rfloor$.

Case 12: : If $y = \lfloor \frac{W-1}{2} \rfloor$, and x is an even number, the codewords of the four VQ indices I, T, L and B are used to reconstruct the block. The equations used to reconstruct the eight pixels are $P_1 = \left\lfloor \frac{c'_{(I,1)}+t_7}{2} \right\rfloor, P_2 = \left\lfloor \frac{c'_{(I,2)}+t_8+l_4}{3} \right\rfloor, P_3 = c'_{(I,3)}, P_4 = c'_{(I,4)}, P_5 = \left\lfloor \frac{c'_{(I,5)}+b_1+l_7}{3} \right\rfloor, P_6 = c'_{(I,6)}, P_7 = c'_{(I,7)}$ and $P_8 = \left\lfloor \frac{c'_{(I,8)}+b_4}{2} \right\rfloor$.

After the above procedure, the compressed pixels have been reconstructed successfully. However, the incompressible pixels have not been recovered yet. The incompressible pixel, $IP(i, j)$, is recovered by the adjacent pixels, e.g., the left pixel LP and the right pixel RP. The equation used to recover the incompressible pixel is:

$$IP_{(i,j)} = \begin{cases} RP, \text{if}\, j \leq W-2, \\ LP, \text{if}\, j > W-2, \end{cases} \tag{1}$$

where (i, j) represents the coordinate of the pixel.

For example, the first VQ index in Fig. 6(c) is 2, and its ID number (x, y) are (1, 1). Since $(x, y) = (1, 1)$, the condition of Case 1 is satisfied. The current VQ index I and its two neighboring indices R and B are mapped into the corresponding codewords, e.g., $I = \{155, 155, 155, 155, 155, 155, 155, 155\}$, $R = \{155, 155, 155, 155, 155, 155, 155, 155\}$, and $B = \{159, 159, 159, 159, 159, 159, 159, 159\}$. The three codewords are used to reconstruct the eight pixels of the first block, i.e., $P_1 = c'_{(I,1)} = 155, P_2 = c'_{(I,2)} = 155, P_3 = c'_{(I,3)} = 155, P_4 = \left\lfloor \frac{c'_{(I,4)}+r_2}{2} \right\rfloor = \lfloor \frac{155+155}{2} \rfloor = 155, P_5 = c'_{(I,5)} = 155, P_6 = c'_{(I,6)} = 155, P_7 = \left\lfloor \frac{c'_{(I,7)}+r_5+b_1}{3} \right\rfloor = \lfloor \frac{155+155+159}{3} \rfloor = 156$, and $P_8 = \left\lfloor \frac{c'_{(I,8)}+b_2}{2} \right\rfloor = \lfloor \frac{155+159}{2} \rfloor = 157$.

Fig. 8(a) shows the first reconstructed block. The procedure of reconstructing the other blocks is the same as the above procedure. Fig. 8(b) shows the reconstructed partial image. After that, all of the incompressible pixels can be recovered effectively by the neighboring reconstructed pixel. For example, the incompressible pixel $IP_{(1,1)}$ in the coordinate (1, 1) is recovered by the right pixel RP, i.e., $IP_{(i,j)} = RP = 155$. Fig. 8(c) shows the reconstructed image.

4. Experimental Results.

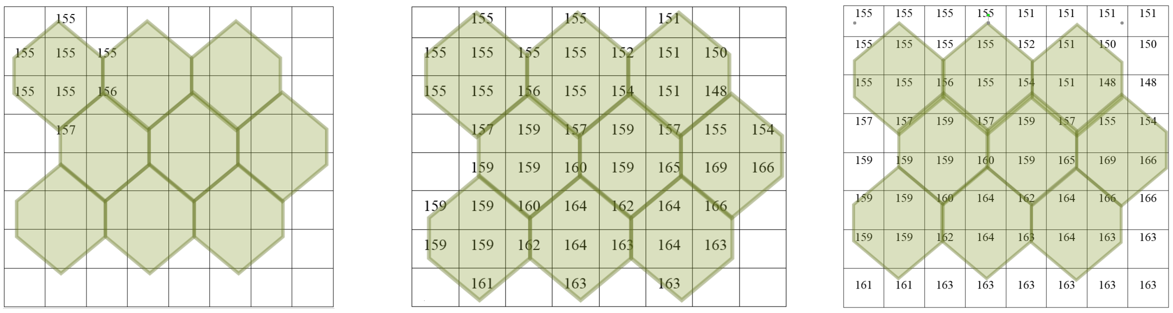
Fig. 9 shows four test images, each of which consists of 512×512 pixels. The four test images are used to generate three codebooks, where the sizes of the codebooks were set to be 256, 512, and 1024. The similarity between the original image and the reconstructed image was measured by peak signal to noise ratio (PSNR)

$$PSNR = 10\log_{10}\frac{255^2}{MSE}, \tag{2}$$

where MSE represents the mean square error between the original image and the reconstructed image, i.e.,

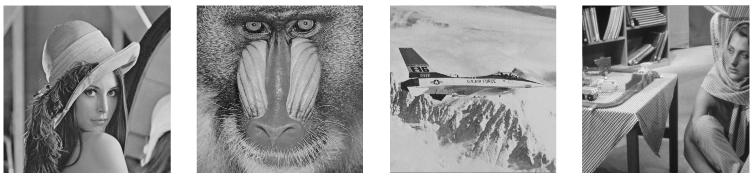
$$NSE = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} (P_{(i,j)} - P'_{(i,j)})^2. \tag{3}$$

The two notations H and W denote the height and the width of the original image, respectively. In addition, $P_{(i,j)}$ and $P'_{(i,j)}$ denote the original pixel and the stego pixel, respectively.



(a) Reconstructed partial image after recovering the first block (b) Reconstructed partial image after recovering each block (c) Reconstructed image

FIGURE 8. Example of decompression procedure



(a) Lena (a) Baboon (c) F-16 (c) Barbara

FIGURE 9. Four test images

Fig. 10 shows that the PSNR value of the reconstructed image and the compression rate (CR) increased as the size of the codebook increased. The experimental results showed that using a larger codebook can exactly match the codeword and the compressed block; thus, the PSNR value of the reconstructed image is high. However, as the number of codewords increases, more bits are required to represent the indices of the codewords, which increases the CR value. Therefore, there is a trade-off between the values of the PSNR and the CR. However, in the proposed method, the size of the codebook can be adjusted freely, depending on the actual needs. Consequently, the proposed method is more practical than the BTC method [1].

Table 1 shows that the PSNR value of the traditional VQ compression method [2] is greater than that of the proposed method. However, there are significant block artifacts in the reconstructed image of the VQ compression method. Fig. 11 shows that the proposed method can effectively reduce block artifacts in the reconstructed images.

Table 2 compares the proposed method with the related methods. The proposed method has a smaller compression ratio (CR) than JPEG, and the PSNR value of the proposed method is higher than that of JPEG. Especially, for the complex image (Baboon), our PSNR value is 6 dB higher than JPEG. This is because both DCT and quantification procedures in JPEG removed the textures of the images, thereby decreasing the quality of the image.

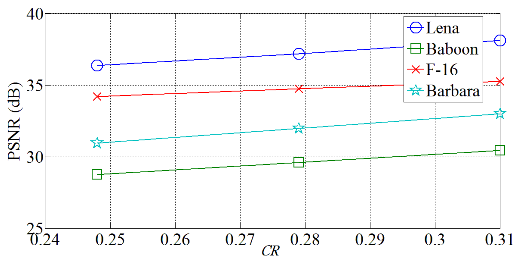
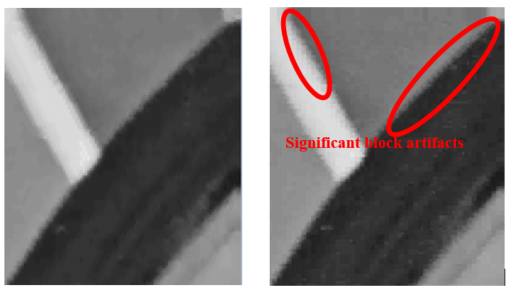


FIGURE 10. Compression rates and PSNR values of the proposed method using different sizes of the codebooks

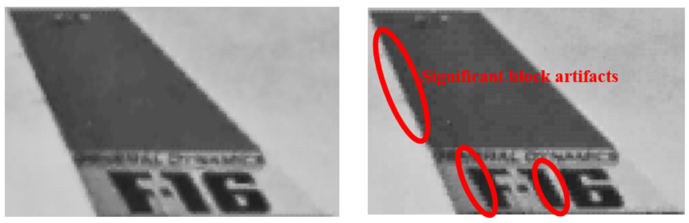
TABLE 1. Comparison between the VQ-based image compression method and the proposed method in terms of PSNR and CR

Image	[2]		Proposed method	
	PSNR	CR	PSNR	CR
Lena	36.93	0.25	36.37	0.248
Baboon	31.41	0.25	28.76	0.248
F-16	37.64	0.25	34.20	0.248
Barbara	32.70	0.25	30.94	0.248



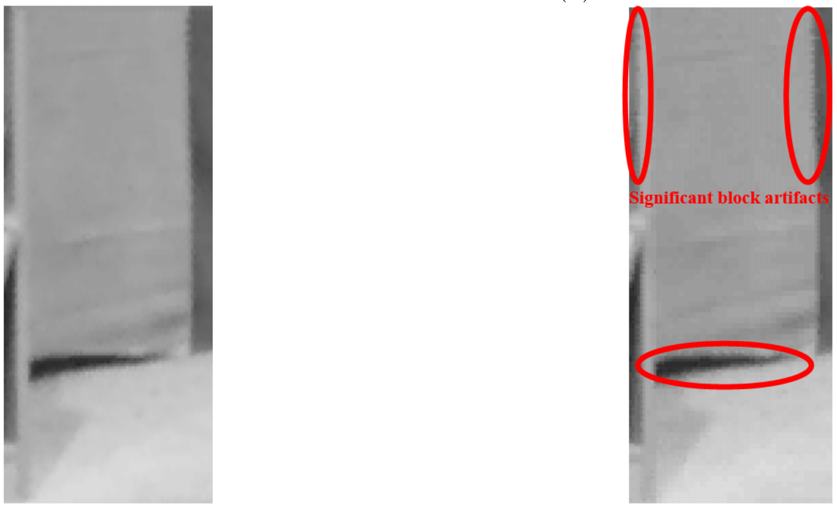
Proposed method VQ method

(a) Part of mirror in Lena



Proposed method VQ method

(b) Vertical stabilizer of F-16



Proposed method VQ method

(c) Door in Barbara

FIGURE 11. Reconstructed images of the proposed method and the VQ method [2]

The PSNR value of Baboon of the proposed method can achieve 28.76 dB, which is 1.7 dB higher than that of the BTC method. This is because the BTC method only classified the pixels in the block into two groups and used the mean of pixels of each group to represent the original pixel, which is unsuitable for encoding more complex block. Different from the BTC method, the proposed method can train a codebook by the features of images, thereby achieving a higher PSNR value.

Table 2. Comparison of the proposed method with the related methods, i.e., JPEG and BTC

Image	JPEG		BTC		Proposed method	
	PSNR	CR	PSNR	CR	PSNR	CR
Lena	32.60	0.33	33.23	0.25	36.37	0.248
Baboon	22.03	0.3	27.01	0.25	28.76	0.248

5. **Conclusions.** In this paper, we proposed a turtle-shell-shaped division method that segments one image into several overlapping blocks. Then, each block is replaced by the VQ index to compress the image. In the decompression phase, the VQ indices are mapped into the corresponding codewords to reconstruct the compressed blocks. It is particularly significant that the overlapping regions between the blocks are reconstructed by means of the codewords of the neighboring VQ indices, thereby avoiding the block artifact. The experimental results showed that the visual quality of the reconstructed image was significantly better than that of the VQ method. In the future, we will try to combine the proposed method with the embedding methods [14] to achieve more applications.

REFERENCES

[1] E. J. Delp and O. R. Mitchell, Image compression using block truncation coding, *IEEE Transactions on Communications*, vol. 27, no. 9, pp. 1335-1342, September, 1979.

[2] N. M. Nasrabadi and R. King, Image coding using vector quantization: A review, *IEEE Transactions on Communications*, vol. 36, no. 8, pp. 957-971, 1988.

[3] Y. Linde, A. Buzo, and R. M. Gray, An algorithm for vector quantizer design, *IEEE Transactions on Communications*, vol. 28, no.1, pp. 84-95, January, 1980.

[4] C. H. Hsieh and J. C. Tsai, Lossless compression of VQ index with search-order coding, *IEEE Transactions on Image Processing*, vol. 5, no. 11, pp. 1579-1582, November, 1996.

[5] F. Ghido and I. Tabus, Sparse modeling for lossless audio compression, *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 1, pp. 14-28, January, 2013.

[6] B. Cheng, C. Ritz, I. Burnett, and X. Zheng, A general compression approach to multi-channel three-dimensional audio, *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 8, pp. 1676-1688, August, 2013.

[7] V. A. Nguyen, D. Min, and M. N. Do, Efficient techniques for depth video compression using weighted mode filtering, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 2, pp. 189-202, February, 2013.

[8] H. C. Kuo and Y. L. Lin, A hybrid algorithm for effective lossless compression of video display frames, *IEEE Transactions on Multimedia*, vol. 14, no. 3, pp. 500-509, June, 2012.

[9] D. A. Huffman, A method for the construction of minimum redundancy codes, *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098-1101, September, 1952.

[10] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 2002.

[11] K. R. Rao and P. Yip, *Discrete Cosine Transform*, New York: Academic, 1990.

[12] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, Image coding using wavelet transform, *IEEE Transactions on Image Processing*, vol. 1, no. 2, pp. 205-221, April, 1992.

[13] C. C. Chang, Y. Liu, and T. S. Nguyen, A novel turtle shell based scheme for data hiding, *Proceedings of the* 10th *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Kitakyushu, Japan, pp. 89-93, August 27-29, 2014.

[14] C. Qin, C. C. Chang, and Y. P. Chiu, A novel joint data-hiding and compression scheme based on SMVQ and image inpainting, *IEEE Transactions on Image Processing*, vol. 23, no. 3, pp. 969-978, 2014.