IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 30, 2018, accepted October 19, 2018. Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2018.2879404

Adaptive Image Self-Recovery Based on Feature Extraction in the DCT Domain

MOHAMED HAMID AND CHUNYAN WANG

Department of Electrical and Computer Engineering, Concordia University, Montreal, QC H3G 1M8, Canada

Corresponding author: Chunyan Wang (chunyan@ece.concordia.ca)

This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada and in part by Concordia University.

ABSTRACT Image self-recovery aims at protecting digital images from partial damage due to accidental or malicious tampering. It is done by generating a reference code that contains the information of the image and embedding the code in the image itself. This code can later be extracted to restore the tampered regions of the image. The reference code must contain sufficient information to ensure a satisfactory reconstruction while being short enough to remain invisible when embedded in the image, which requires efficient extraction and adaptive encoding of the image information. To this end, we introduce a method for extracting local features in the DCT domain, in which the locations of the three DCT peaks, i.e., the DCT coefficients with the highest magnitudes, are examined to distinguish 13 texture profiles differing in the number of edges, edge orientations, and combinations of the two. Applying this method, we propose an adaptive image self-recovery algorithm. The DCT peaks are used to identify local texture patterns, and the bit allocation is made adaptive at hierarchical levels: 1) the texture blocks get more bit allocation than the smooth blocks; 2) the blocks having texture patterns appearing more frequently in the image are encoded with more precision; and 3) in each texture block, the highest DCT peak is assigned more bits than the remaining encoded coefficients. Hence, the encoding process is not only adaptive to the levels of variations across blocks but also to the local texture patterns. The proposed algorithm generates a reference code short enough to be embedded very comfortably in a single-least-significant-bit (LSB) plane, compared to $2 \sim 3$ LSB planes often found in literature. Since the reference code contains all the critical image information in a compact form, the quality of the reconstructed images is as good as those produced by significantly longer reference codes.

INDEX TERMS Image features extraction and presentation, DCT peaks, image self-recovery, adaptive bit allocation.

I. INTRODUCTION

The widespread use of the internet and the effortless accessibility to photo-editing tools has made digital images vulnerable to manipulations and malicious tampering. Image self-recovery is an approach to image protection wherein image information is encoded and then embedded in the image itself. If the image is tampered with, the altered regions can be recovered using the information extracted from the untampered parts of the image.

The embedded code, referred to as the reference code, must contain sufficient information to allow a satisfactory restoration of the image with sufficient details. It is often embedded in the least significant bit (LSB) planes of the image in the spatial domain. To make the embedded code invisible, it should be short enough to be accommodated in a single LSB plane, which is difficult to achieve in practice.

A common approach in self-recovery algorithms is to encode the frequency components of image blocks, as the visual information is more compact in the frequency domain than in the spatial domain. In the methods reported in [1]–[5], the same number of bits is non-discriminatively assigned to encode the frequency components of every image block. As the signal complexity varies among image blocks, the length of the reference code can be effectively reduced by making the bit allocation related to some statistics of gray-level variations. In the methods reported in [6]–[8], the blocks are categorized according to the level of signal variations, and the blocks having a higher level of variations get more bits allocated. In the algorithm reported in [9], the smoothest blocks are skipped entirely in the encoding process to minimize the length of the reference code and an inpainting process is used to restore these blocks.

The reference codes produced by many reported self-recovery algorithms, including those mentioned above, are too long for single LSB embedding and some even require 3 LSB planes, which affects the visual quality of the code-embedded images. To make the code short enough to fit within a single LSB plane while containing sufficient information to reconstruct visible image features, the image elements in each block should be encoded with variable code lengths. In other words, the bit allocation must be adaptive to the local image patterns of individual image blocks. To this end, the self-recovery algorithm needs to include a process that efficiently extracts and compactly represents the local features.

The discrete cosine transform (DCT) is one of the simplest methods to obtain a compact representation of the image in the frequency domain, and therefore it is commonly incorporated in image processes. It is thus deemed efficient to extract image features using the DCT coefficients and some attempts to do so have been reported. In [10], the edge orientations are identified after calculating a partial inverse DCT, which is computationally intensive. In [11] and [12], a comparison of the amplitudes of the two DCT coefficients with the lowest frequencies is used to detect edge orientations, while in [13], more coefficients are included in the detection. In [14], the averages of the coefficients located in the first row, the first column and the diagonal line of the DCT matrix are used for the same purpose. These reported methods rely on predefined sets of coefficients. However, the critical information of many patterns may be carried by DCT coefficients that are not included in the predefined sets. Moreover, these methods usually focus on extracting the information of edge orientations only. Generally, it is more challenging to accurately classify the visible patterns in image blocks and represent them with a minimum amount of data.

In this paper, we propose a DCT-based feature extraction method that can be used to detect the texture pattern in an image block by examining the locations of the DCT peaks, i.e., the DCT coefficients with the highest amplitudes. This method allows us to efficiently extract the feature information of specific patterns, besides the levels of variations, in individual image blocks and develop an adaptive image self-recovery algorithm that generates reference codes short enough to be embedded in single-bit LSB planes. The short code length is achieved by adaptively encoding the high-density feature information extracted from each image block. The feature extraction method is presented in Section II. The proposed image self-recovery algorithm is described in Section III and the simulation results are discussed in Section IV.

II. DCT PEAKS AND IMAGE FEATURE REPRESENTATION

An image usually consists of flat regions that contain little to no variations, and texture regions that contain significant gray level variations. In the DCT domain, the DC coefficient in the DCT matrix of a texture block indicates the average gray level of the block, while the variations are represented by the AC coefficients. Although image patterns in blocks sized 8×8 pixels are usually simple, their spectrum may extend to a wide range of frequencies. However, the AC coefficients with the highest magnitudes, referred to as the DCT peaks, indicate the most perceivable gray level variations of a texture block. In other words, the DCT peaks carry the most critical information of the block. Hence, one may use a very small number of DCT peaks, instead of the complete set of DCT coefficients, to represent an image block without losing the most important image information required in a defined image process.

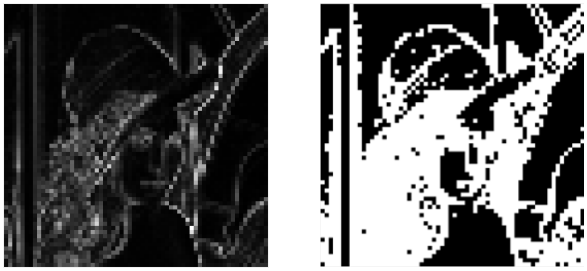
In a flat block, the highest AC coefficients merely reflect insignificant gray level fluctuations and the image signal is determined by the DC coefficient. Hence the proposed technique to use the DCT peaks for image feature representation is applied solely to the texture blocks. The process involves two steps: identifying the texture blocks of the image, and then analyzing these blocks.

A. IDENTIFICATION OF THE TEXTURE BLOCKS

The presence of image texture is indicated by visible gray level variations that can be detected by using various techniques in the spatial domain [15], [16] or the AC coefficients in the frequency domain. In this work, the highest DCT peak and the DC coefficient in each of the DCT matrices are used to identify texture blocks.

The highest DCT peak represents the most significant gray level variations of a block. However, whether these variations are perceived as texture patterns or as insignificant gray level fluctuations, depends on the DC component. That is because the human visual system (HVS) is less sensitive to gray level variations in a brighter background, i.e. higher gray level mean value [17]. Hence, the ratio of the highest DCT peak magnitude to its DC coefficient, denoted as *Peak/DC*, can be effectively used to identify blocks with visible patterns.

As an example, the identification method described above has been applied to the image Lena. In Fig. 1(a), each block is represented by the value of its *Peak/DC* ratio. The white blocks in Fig. 1(b) indicate the locations of the blocks that have *Peak/DC* ratios in the upper 50%, which are considered to be texture blocks.



(a) (b)

FIGURE 1. Identification of texture blocks by using the *Peak/DC* ratios. (a) *Peak/DC* map of Lena, and (b) Binary map with the locations of the texture block shown in white.

The identification of the texture blocks by means of the *Peak/DC* ratios is effective and may be the simplest. In general, a normalized DCT coefficient can be used to measure the significance of gray level variations, yet the normalization can be done differently to suit various processing purposes. For instance, instead of the highest DCT peak, one can normalize the average value of the three highest peaks by the value of the DC coefficient. One can also determine a threshold based on the distribution of the *Peak/DC* ratios of the image blocks and apply it to identify texture blocks.

B. FEATURE REPRESENTATION

The most perceivable image features of a texture block of 8 × 8 pixels are determined by the locations and the amplitudes of the most significant DCT coefficients in the 8 × 8 DCT matrix. Although the DCT peaks vary from block to block, the distribution of the DCT peak locations of the texture blocks of an image indicates the effective range of the ''local'' frequencies, i.e. frequencies in the blocks.

Fig. 2 shows the distribution of the highest DCT peaks in the individual image blocks of 200 test images, excluding the 50% of the blocks with the lower *Peak/DC* ratios. It can be observed that the highest DCT peaks of approximately 65% of the blocks are found at locations #1 and #2 in the DCT matrix, which correspond to the lowest two frequencies. However, a notable 30% of the blocks have their highest peaks at locations #3 ∼ #9, demonstrating that these frequency components in that medium range carry significant visual information and are thus non-negligible. The overwhelming majority (99%) of the peaks are located in the top-left 4 × 4 section of the 8 × 8 DCT matrix. The coefficients outside this 4 × 4 section are mostly of high spatial frequencies and have a minor impact on the visibility of the block patterns. Thus, it is reasonable to focus the analysis on the top-left 4 × 4 coefficients in an effort to simplify the process.

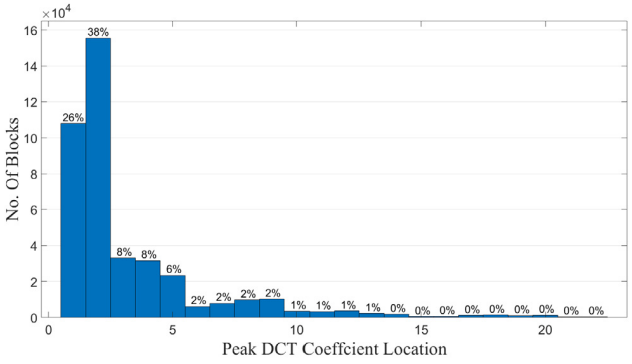


FIGURE 2. Distribution of the locations of the highest peaks in the DCT matrices of 200 images. The x-axis is the location numbered in the zigzag order used in JPEG compression.

Image features are made of edges and each edge is characterized by its orientation, magnitude and offset. The following observations relate the DCT peaks to these edge characters in the spatial domain:

- *The horizontal, vertical and diagonal edge orientations can be represented by the two highest peaks among the AC coefficients*. There are two cases. In the first case, the peaks are both located in the first row or the first column of the DCT matrix, and the pattern appearing in the image block must have a dominantly horizontal or vertical edge orientation. Some of such patterns and their DCT matrices are shown in Figs. 3(a) & (A), (b) & (B), (d) & (D) and (e) & (E). In the second case, a diagonal edge, as an example shown in Fig. 3(c), is represented by the two peaks of comparable magnitudes, one being in the first row and the other in the first column, as shown in Fig.3(C), because a diagonal edge can be seen as the result of a gray level change in the vertical direction superimposed with one in the horizontal direction.

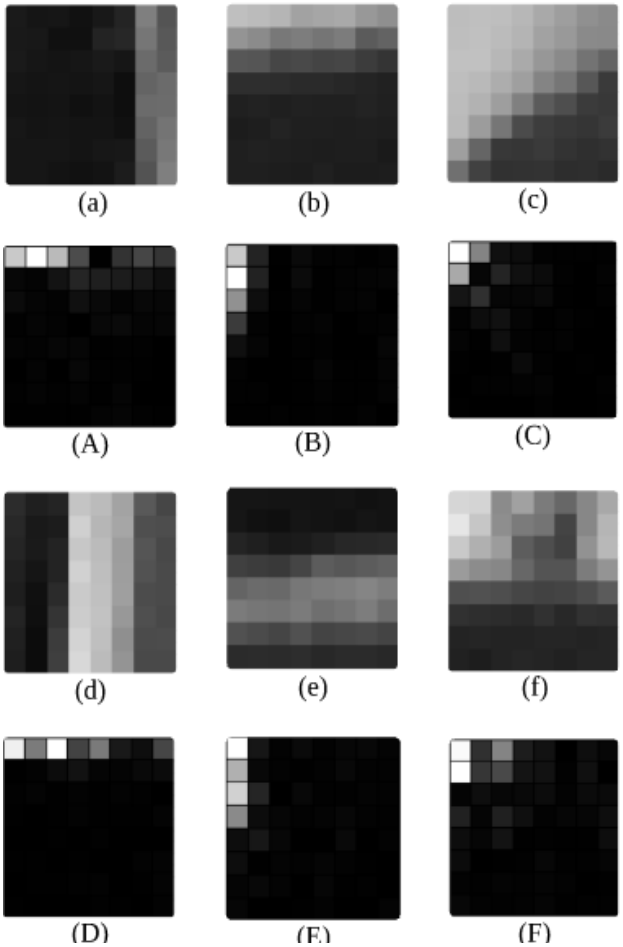


FIGURE 3. (a) – (f) Examples of 8 × 8 image blocks with simple patterns (A) – (F) DCT matrices of the blocks.

- *The number of parallel edges in the dominant orientation is indicated by the location of the highest peak*. In case of multiple parallel edges, the highest peak will be found at a higher spatial frequency location such as the examples shown in Fig. 3 (d) and (e).
- *Edge offset is indicated by the second and/or third highest peak found in the same row or column as the*

highest peak in the DCT matrix. Such cases are illustrated in Fig. 3 (a) and (b).

- *Multiple edges of different orientations are related to the DCT peaks dispersed in the 2-D matrix*. Among the peaks, the highest indicates the most noticeable direction of gray level variation, while the second and/or third are in higher frequencies in a different row or column to indicate the direction of the second grey level variation. An example is shown in Fig. 3 (f).

In summary, among the DCT peaks, the first peak, i.e. the highest peak, indicates the dominant edge orientation as well as the number of parallel edges in that orientation, while the second and third peaks indicate the offset of the gray level pattern and/or the presence of visible edges in a direction different from the dominant one.

As mentioned previously, the effective frequency range for the feature representation by the DCT peaks is given by the top-left 4×4 locations in the DCT matrix. The highest 3 DCT peaks can be located anywhere within this range. To simplify the process, the 15 AC coefficients in the top-left 4×4 locations are divided into 7 groups as shown in Fig. 4. The coefficients in each group have some common features. For example, each of the 3 coefficients of group "d1" represents a multiple-diagonal-edge component.

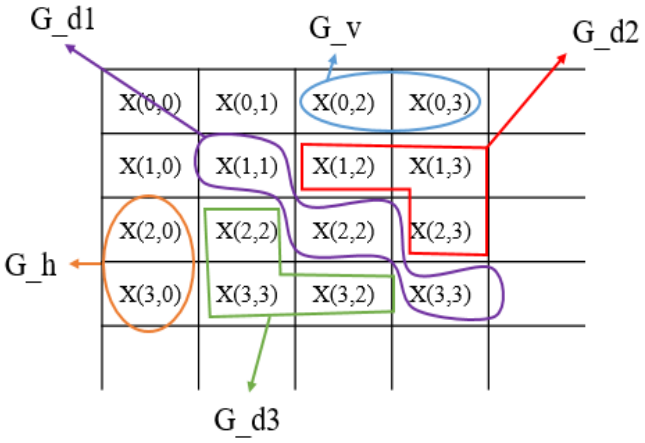


FIGURE 4. DCT coefficients' groups. The coefficient X(0, 1) makes one group, so does X(1, 0).

As image blocks of 8×8 pixels are small enough to contain only simple patterns, a small set of thirteen profiles, shown in Table 1, has been defined to represent the possible texture patterns in image blocks. A texture block can be classified into one of these profiles according to the locations of the highest 3 peaks in the DCT matrix as illustrated in Table 2.

The effectiveness of this representation method can be tested by comparing the pattern reconstructed by using only the 3 DCT peaks of a block with the original pattern constituted by the full set of 8×8 DCT coefficients. In case of simple image patterns shown in Fig. 3 (a)∼(e), eliminating all AC coefficients other than the 3 peaks reduces the sharpness of the edges, but the texture pattern remains the same. Consider the case of the complex texture block shown in Fig. 3 (f) as an example. The block and its full 8×8 DCT matrix are shown in Fig. 5 (a). The pattern in Fig. 5(b) is generated by using the DCT coefficients at the top-left 2×2 locations, and it differs greatly from the original pattern as the critical higher frequency information is excluded from the process. In Fig. 5 (c), the range is extended to include all the top-left 4×4 coefficients. The pattern in Fig. 5(d) has similar features with those in Fig. 5(c), however it is generated by using only the 3 highest peaks in addition to the DC coefficient,

TABLE 1. Defined profiles.

Profile	Description	Examples
V1	Vertical edge	
H1	Horizontal edge	
V2	Parallel vertical edges	
H2	Parallel horizontal edges	
D1	Vertically dominant diagonal edge	
D2	Parallel vertically dominant diagonal edges	
D3	Horizontally dominant diagonal edge	
D4	Parallel horizontally dominant diagonal edges	
D5	Parallel diagonal edges	
T1	Horizontal T shape	
T2	Vertical T shape	
Rec	Rectangular shape	
Cross	Cross shape	

TABLE 2. Classification according to the highest three DCT peaks.

1st Peak	2nd Peak	3rd Peak	Profile
$X(0,1)$	$X(1,0)$	×	D1
	G_v	$X(1,0)$	D1
		G_h or G_d3	T1
		otherwise	V1
	G_d1 or G_ d2	×	D2
	G_h or G_d3	×	T1
G_v	First row	×	V2
	G_d1 or G_d2	×	D2
	G_h or G_d3	×	Cross
$X(1,0)$	G_h	$X(0,1)$	D3
		G_v or G_d3	T2
		otherwise	H1
	$X(0,1)$	×	D3
	G_d1 or G_d3	×	D4
	G_v or G_d2	×	T2
G_h	First column	×	H2
	G_d1 or G_d3	×	D4
	G_v or G_d2	×	Cross
G_d1	G_d1	×	D5
	$X(0,1)$	$X(1,0)$	Rec
		otherwise	D2
	$X(1,0)$	$X(0,1)$	Rec
		otherwise	D4
	G_v or G_d2	G_h or G_d3	Cross
		otherwise	D2
	G_h or G_d3	G_v or G_d2	Cross
		otherwise	D4
G_d2	First row or G_d1	×	D2
	First Column or G_d3	×	Cross
G_d3	First row or G_d2	×	Cross
	First Column or G_d3	×	D4

* × represents don't care conditions

ignoring the remaining AC coefficients. Clearly, the feature information of the pattern is concentrated in the three peaks.

It is evident that the most critical features of an image block can be represented, with sufficient accuracy, by the DCT peaks of the block. This approach can lead to an effective and computationally simple image feature extraction or high ratio compression procedures. The use of the DCT peaks can also be expanded by including more coefficients, or by increasing the number of profiles. In the following section, the application of this method in image self-recovery is presented.

III. IMAGE SELF-RECOVERY ALGORITHM

The main challenge in image self-recovery is to minimize the reference code in order to maintain good visual quality of the code-embedded image, while achieving good restoration of the lost image details. To achieve that, we utilize the feature extraction method presented in Section II to develop an adaptive image self-recovery algorithm, in which the bits are allocated according to the texture patterns of individual image blocks. The general scheme of the proposed image self-recovery algorithm is illustrated in Fig. 6. Detailed description of the algorithm is found in the following sub-sections.

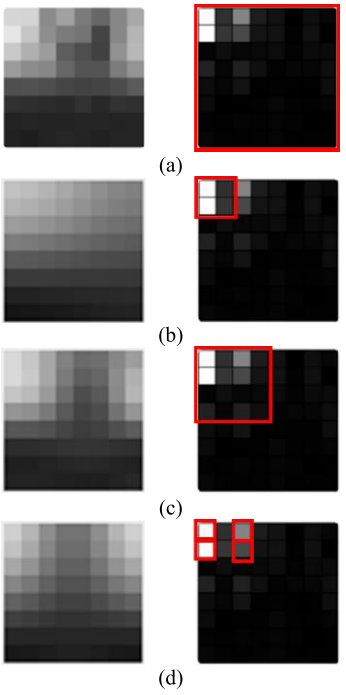


FIGURE 5. Image patterns and the DCT coefficients, specified by the red-frames, used to generate them.

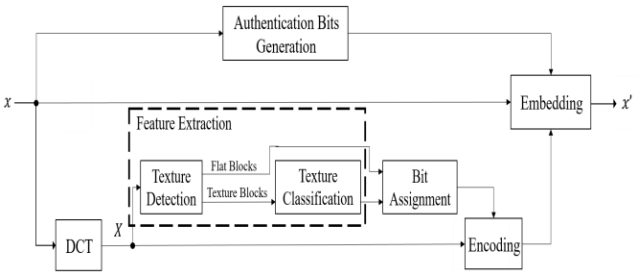


FIGURE 6. General scheme of the proposed image self-recovery algorithm.

A. FEATURE EXTRACTION AND BIT ALLOCATION

The local feature extraction in the proposed self-recovery algorithm is performed on the DCT coefficients of each image block to identify the most critical DCT coefficients representing the most perceivable features of the image blocks, so that they can then be encoded with a minimized code length. As shown in Fig. 6, the procedure consists of three steps:

a) Detecting texture blocks by applying the method described in Section II. If the *Peak/DC* ratio of an

image block is in the upper part of the range, it will be identified as a texture block.

b) Classifying the texture block to one of the 13 profiles illustrated in Table 1, based on the locations of the 3 DCT peaks in the 8 × 8 DCT coefficient matrix, as presented in Table 2.

c) Assigning bits to encode the DCT coefficients by applying the bit allocation map corresponding to the texture profile identified in the previous step.

There are 2 key characteristics in the proposed self-recovery algorithm. The first is that, based on the locations of the 3 DCT peaks in the DCT matrix, only a very small part of the 8 × 8 DCT coefficients is chosen to be encoded and embedded to represent the extracted features. The other characteristic is the variable code lengths. The DCT coefficients determining the most perceivable features, such as the DC component and the highest peak, are assigned the highest number of bits, i.e., 8 bits. In this way, the DCT coefficients having a better concentration of the image information, no matter which frequency ranges they fall within, will be encoded with a better precision to preserve critical image details, while the overall code length of the image is minimized.

The DC coefficient is the most crucial element in the DCT matrix of a block, whether it has textures or not, as it determines the average gray level. Hence 8 bits are assigned to it for the best precision in the encoding process. In a flat block, only the DC coefficient is encoded to represent the signal of the block, and the other coefficients are ignored.

In case of texture blocks, to make the code length adaptive, a set of bit allocation maps has been designed for the 13 texture profiles. As mentioned previously, the profile of a texture block is identified by the locations of the highest three peaks, as shown in Table 2, and all of them are found in the top-left quarter of the 8 × 8 DCT coefficient matrix, as previously discussed. The bit assignment is to map bits to the 4 × 4 locations of this quarter. Hence the bit allocation maps, shown in Fig. 7, are 4 × 4 matrices.

The 13 profiles defined in Table 1 are divided into 3 groups according to the spatial frequencies of their texture features. The first group consists of the profiles "V1", "H1", "D1", and "D3" shown in Table 1. Each block in this group features a single-edge, and thus the first DCT peak is found in the locations indicated in Fig. 4 as X(0, 1) or X(1, 0), i.e., the first or second in the zigzag order. The majority of the highest peaks (65%) are found in one of these two locations as noted previously from Fig. 2. Therefore, these four profiles are the most frequently appearing, i.e., the most visible, in images, and their DCT peaks merit the most precise encoding.

In case of the profiles "V1" and "H1", the first peak is dominant and the other 2 are harmonics. The highest number of bits, i.e., 8, is assigned to the location X(0, 1) in case of "V1", or X(1, 0) in case of "H1". 6 bits are allocated to second DCT peak and 5 bits to the third peak, as shown in Fig. 7. Please note that, unlike the first peak, of which the location is certain, the second peak can be found in one of

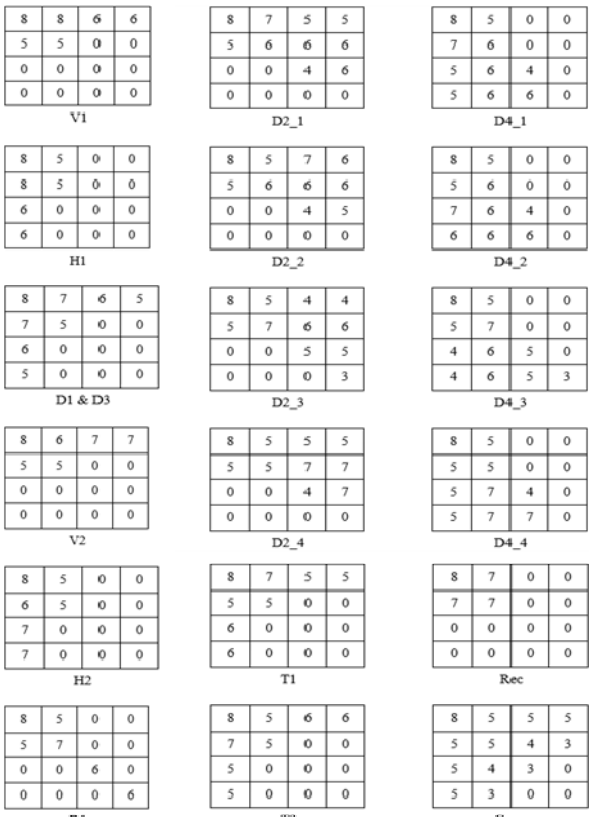


FIGURE 7. Bit allocation maps.

two possible locations, e.g., either X(0,2) or X(0,3) in case of "V1", depending on the exact shape of the edge; therefore, 6 bits are assigned to both locations.

The profiles "D1" and "D3" are characterized by the first and second DCT peaks having comparable magnitudes and located at X(0, 1) and X(1, 0). As both DCT peaks are equally important in determining the diagonal edge of the block, both locations have an equal assignment of 7 bits. The location of the third peak is less certain, and thus 6 bits is assigned to the locations where it is more likely to appear and 5 bits to the less-likely but possible locations.

The second group consists of the profiles "V2", "H2", "D2", "D4", "D5", "T1" and "T2", featuring parallel edges, and thus their DCT peaks are found in higher spatial frequency locations. These profiles appear less frequently than those of the first group but are still part of the visible features. Hence, 7 bits are assigned to the locations where the first DCT peak may appear; and 6 bits and 5 bits are assigned to the possible locations of the second peak and the third peak, respectively.

In case of "D2" or "D4", the first peak can appear in several locations, as indicated in Table 2, corresponding to different edge orientations. Accordingly, there are 4 different bit assignment maps for each of these two profiles, as shown in Fig. 7. Therefore, for "D2" or "D4", 2 bits are added to the beginning of the code to specify the assignment map used for the block.

The two remaining profiles, namely "Rec" and "Cross", belong to the third group. As the profile "Rec" can be seen as superimposed "V1", "H1" and "D5", the three DCT peaks of comparable magnitudes are located at X(1,1), X(0, 1) and X(1, 0), and 7 bits are assigned to each of the three locations. The locations of the three DCT peaks of "Cross" are less certain. They can appear in several sets of locations, as shown in Table 2, and the DCT coefficients in all these locations need to be encoded. Since this profile appears less frequently than the others in an image and its detail is less perceivable, the peaks can be encoded with less precision. Thus, the numbers of bits assigned to the locations where the three peaks can appear are 5, 4 or 3.

To summarize, in the proposed image self-recovery algorithm, the feature information is extracted and represented by the DCT peaks. To encode the DCT coefficients with a minimum number of bits while keeping a good precision, the bit assignment to an image block is made adaptive to its texture profile. The number of bits assigned to a flat block is 8, and that to a texture block is between 29 and 60, as shown in the bit maps presented in Fig. 7. As 50% of the blocks in an image are considered flat in this implementation, the "average" number of bits per block is 19 in the best case and 34 in the worst case. Considering that the most frequently appearing texture blocks have profiles "V1" & "H1" and each of these profiles is allocated 38 bits, the average number of bits per block is below 30. Hence the reference code is short enough for one-bit LSB embedding.

B. ENCODING AND EMBEDDING

The code to be embedded in the input image is composed of the reference code R, the profile map M, and the authentication bits H.

- The reference code R is generated from the DCT coefficients of the image blocks.
- The profile map M specifies the profile of each image block. There are 13 profiles in the image blocks, and thus 4 bits are used in each block to indicate it. The profile map M of the image is crucial in decoding the reference code, and it must be embedded multiple times in various locations of the image to ensure the survival of a complete profile map after tampering.
- The authentication bits H are used to detect tampering and to localize the tampered blocks of the image upon reception. In the proposed algorithm, the authentication bits for each image block are generated using the MD5 hashing method in the same way as in [9]. The generated authentication code is 16 bits per block.

If N denotes the total number of the blocks in a cover image, the number of bits of 1-LSB plane will be $64N$. The length of the reference code R in the proposed algorithm is between $19N$ and $34N$ bits. The length of the authentication code H is $16N$ bits, and 4 jN bits will be used for the profile map M, if it is embedded j times in the image. In the worst-case scenario, i.e., the length of R being $34N$ bits, the total code length will be $34N + 16N + 4\ jN$ bits. If $j = 3$, this code length will be $62N$ bits, which is short enough for 1-LSB embedding.

The embedding procedure comprises the following steps:

1. The reference code R is divided into N segments, each containing S bits, to form the code R' expressed as:

$$R' = \left\{R'_{\mathrm{n}} : n = 1, 2, 3, \ldots, N\right\}, \tag{1}$$

$$S = \frac{\text{Length of } R}{N}. \tag{2}$$

The index number n is mapped to m to reorder R', according to a secret key k. The new reordered code is referred to as R''. The mapping is performed according to the following mapping function:

$$m = \begin{cases} (n+k)\ mod\ N, & if\ n+k \neq N \\ N, & if\ n+k = N \end{cases} \tag{3}$$

where *mod* refers to the modulus operation. The value of S is important for decoding since the reference code length will vary from one image to another. As the length of the reference code R is between $19N$ and $34N$, S can be represented by $S = S - 19$, and S is encoded in 4 bits.

2. The profile map M is embedded j times and to do so, S' is concatenated to the beginning and the end of M and then, if $j = 3$, a new profile map M' is generated as expressed below.

$$M' = \left\{S', M, S', S', M, S', S', M, S'\right\}. \tag{4}$$

Then, M' is divided into N segments to form M''.

3. In the n^{th} image block x_n, its authentication code H_n, the n^{th} segment of M''_n and the n^{th} segment of the R''_n are embedded into the LSBs of the pixel values to produce the code-embedded block x'_n.

The procedure to retrieve the reference code is simply the inverse of that to embed the code.

IV. SIMULATION RESULTS

The effectiveness of the proposed algorithm has been assessed in terms of the length of the reference code, the visual quality of the image after embedding the code, and that of the reconstructed image. It has been applied to 50 images from the BOWS2 database [18]. The size of the images used in the simulation is 512×512 and the block size is 8×8. The simulation results have been presented as part of the Master's thesis of Mohamed Hamid.

The average reference code length generated by applying the proposed algorithm to the 50 images is 116,039 bits, i.e. about 44% of 512×512 bits of the one-bit LSB plane and the remaining 56% of the plane is available for the authentication code and the profile map. In this case, the profile map can be embedded multiple times in the image to secure a quality recovery.

Because of the one-bit LSB code embedding, the code-embedded images resulting from the proposed self-recovery

algorithm have a high visual quality of 51.64 dB, while those using 2-bit LSB can hardly go above 45 dB.

To give an example of reconstruction using the proposed algorithm, the images of Lena and Boat, shown in Fig. 8, where partially tampered with as shown in Fig. 9, and the reconstructed images are presented in Fig. 10. The tampered parts were reconstructed successfully. Evidently, despite its short length, the reference code contains sufficient image feature information for a reconstruction of good quality.

(a) (b)

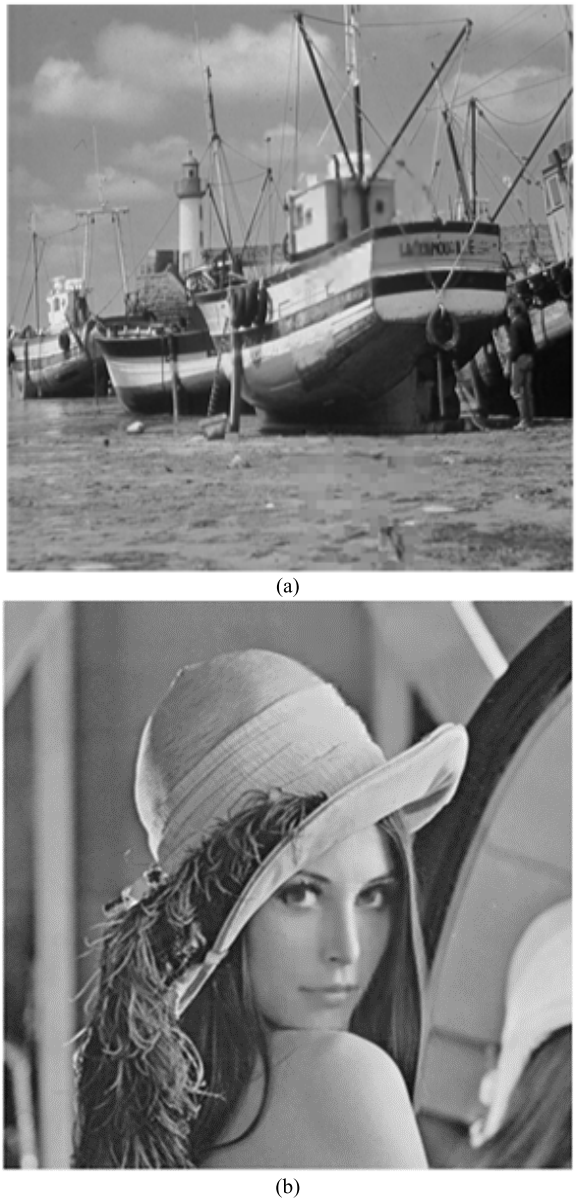
FIGURE 8. Original Images. (a) Boat. (b) Lena.

(a) (b)

FIGURE 9. Tampered Images. (a) Boat. (b) Lena.

Fig. 11 illustrates a comparison of the lengths of the reference codes resulting from different algorithms and the PSNR values of the images reconstructed with the codes at the tampering rate of 20%. The reference code generated by the proposed algorithm has the shortest length. It is about 30% of that reported in [6], the shortest among the compared codes, and it is only 17.6% of that found in [2]. The quality of the images reconstructed using the significantly shorter reference codes is, however, comparable to those generated by other algorithms, as shown in Fig. 11. It confirms that the reference code generated by using the proposed algorithm represents, at a much higher density, the image information that is the most critical for the reconstruction.

The performance of the proposed algorithm has been assessed at a range of tampering rates from 10% to 50%. The results, in terms of PSNR of the reconstructed images, are illustrated in Fig. 12. They confirm that the proposed algorithm is fully functional in the entire tampering range, maintaining a reconstruction quality good enough to be compared with those generated with much longer reference codes.



(a)

(b)

FIGURE 10. Images reonstructed with the proposed algorithm. (a) Boat. (b) Lena.

The PSNR value given by the proposed algorithm is approximately at the median level across this range, while other algorithms may not be applicable when the tampering range is above 30%. It is also observed that only the algorithm in [6] gives a higher PSNR in the entire tampering range, but it is obtained at the expense of a reference code that is 3.4 times longer and in turn a significantly lower visual quality of the code-embedded images.

Table 3 gives an overview of the performance metrics. Overall, it demonstrates that the proposed algorithm generates significantly shorter reference codes than the compared methods, which consequently result in a much higher visual quality in the code-embedded images. As the short codes are generated by adaptively encoding the high-quality features

TABLE 3. Comparison of the code lengths and the PSNR of reconstructed and code-embedded images.

Method	Reference Code Length (Kb)	Average PSNR (dB)					
		Code-Embedded Images	Reconstructed Images at Different Tampering Rates				
			10%	20%	30%	40%	50%
[4]	524	44.15	40.5	40.5	40.5	N/A	N/A
[6]	393	44.15	44	44	39	37	34
[3]	655	37.92	36.4	36.4	36.4	36.4	36.4
[2]	659	37.92	37.2	34.8	29.1	28.1	27
Proposed	**116**	**51.64**	**41.38**	**38.34**	**36.75**	**35.51**	**30.59**

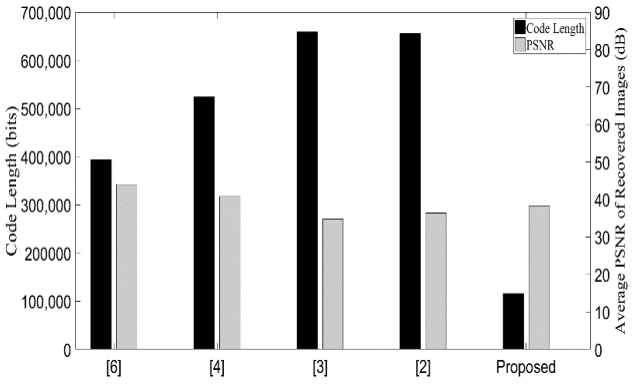


FIGURE 11. Comparison of the code lengths and reconstruction quality at a tampering rate of 20%.

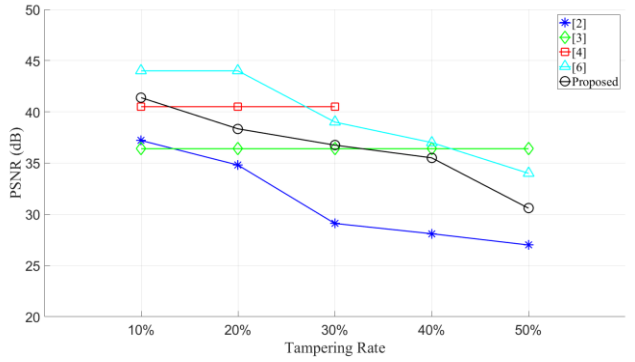


FIGURE 12. Reconstruction quality at different tampering rates.

extracted from the original image, they result in an above-average reconstruction quality.

V. CONCLUSION

An adaptive image self-recovery algorithm has been proposed in this paper. To minimize the length of the reference code without losing visible image details, the encoding has been made adaptive to the local features in individual image blocks. To better obtain and represent the local features, a new method for feature extraction in the DCT domain has been developed. In this method, the image blocks are first divided into flat and texture blocks. The texture blocks are identified by having higher *Peak/DC* ratios. Then, the patterns in the texture blocks are classified into 13 profiles containing different combinations of single and multiple edges with different orientations, and the locations of the 3 highest peaks in the DCT matrix are used to identify the texture pattern, out of the 13 profiles, in each block.

The proposed feature extraction method has been applied in the design of the proposed image self-recovery algorithm. The local features information in each texture block is represented by the DCT peaks, and the bit allocation for encoded is adapted to signal characters at different levels: From the perspective of the entire image, more bits are assigned to texture blocks than flat blocks; within the texture blocks, more bits are allocated to encode the DCT peaks in the blocks having the texture patterns appearing more frequently; and among the DCT peaks in a texture block, the highest peak is assigned with more bits than the others. In this way, all the DCT coefficients critical to visible image details are encoded with good precision, while keeping the entire reference code short. The test with a good number of images has shown that the number of bits of a reference code generated by the proposed algorithm is approximately 44% of the number of pixels in each of the test images. Hence these reference codes have been, very comfortably, embedded within 1 LSB plane. In contrast, 2-3 LSB planes are required by many algorithms found in literature. Consequently, the code-embedded images produced by the proposed algorithm have a superior visual quality. It has also been confirmed that the short reference codes carries critical information determining the most visible image patterns and, as a result, the quality of the reconstructed images is as good as those produced by codes that are 3-6 times longer.

The results of the image self-recovery algorithm prove the efficiency of the feature extraction/representation method proposed in this paper. The method can be applied in various applications where a compact representation of local features is required for adaptive processing. Moreover, the good image reconstruction results confirm the effectiveness of the adaptive bit allocation approach used in the self-recovery algorithm. This approach can be useful in many applications where compressed versions of images or video frames are required.

REFERENCES

[1] X. Zhang, S. Wang, and G. Feng, "Fragile watermarking scheme with extensive content restoration capability," in *Proc. Int. Workshop Digit. Watermarking*, 2009, pp. 268–278.
[2] X. Zhang, Z. Qian, Y. Ren, and G. Feng, "Watermarking with flexible self-recovery quality based on compressive sensing and compositive reconstruction," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 4, pp. 1223–1232, Dec. 2011.
[3] P. Korus and A. Dziech, "Efficient method for content reconstruction with self-embedding," *IEEE Trans. Image Process.*, vol. 22, no. 3, pp. 1134–1147, Mar. 2013.
[4] S. Sarreshtedari and M. A. Akhaee, "A source-channel coding approach to digital image protection and self-recovery," *IEEE Trans. Image Process.*, vol. 24, no. 7, pp. 2266–2277, Jul. 2015.
[5] S. Sarreshtedari, M. A. Akhaee, and A. Abbasfar, "Source–channel coding-based watermarking for self-embedding of JPEG images," *Signal Process., Image Commun.*, vol. 62, pp. 106–116, Mar. 2018.
[6] P. Korus and A. Dziech, "Adaptive self-embedding scheme with controlled reconstruction performance," *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 2, pp. 169–181, Feb. 2014.
[7] C. Qin, C.-C. Chang, and P.-Y. Chen, "Self-embedding fragile watermarking with restoration capability based on adaptive bit allocation mechanism," *Signal Process.*, vol. 92, no. 4, pp. 1137–1150, 2012.
[8] C. Qin, C.-C. Chang, and K. N. Chen, "Adaptive self-recovery for tampered images based on VQ indexing and inpainting," *Signal Process.*, vol. 93, no. 4, pp. 933–946, Apr. 2013.
[9] Z. Qian and G. Feng, "Inpainting assisted self recovery with decreased embedding data," *IEEE Signal Process. Lett.*, vol. 17, no. 11, pp. 929–932, Nov. 2010.
[10] H. S. Chang and K. Kang, "A compressed domain scheme for classifying block edge patterns," *IEEE Trans. Image Process.*, vol. 14, no. 2, pp. 145–151, Feb. 2005.
[11] B. Shen and I. K. Sethi, "Direct feature extraction from compressed images," *Proc. SPIE*, vol. 2670, pp. 404–415, Mar. 1996.
[12] M. Eom and Y. Choe, "Fast extraction of edge histogram in DCT domain based on MPEG7," in *World Acad. Sci., Eng. Technol.*, vol. 1, no. 9, pp. 1405–1408, 2005.
[13] H. Li, G. Liu, and Y. Li, "An effective approach to edge classification from DCT domain," in *Proc. Int. Conf. Image Process.*, Sep. 2002, pp. I-940–I-943.
[14] S.-J. Park, G. Jeon, and J. Jeong, "Deinterlacing algorithm using edge direction from analysis of the DCT coefficient distribution," *IEEE Trans. Consum. Electron.*, vol. 55, no. 3, pp. 1674–1681, Aug. 2009.
[15] C.-H. Lin, C.-W. Liu, and H.-Y. Chen, "Image retrieval and classification using adaptive local binary patterns based on texture features," *IET Image Process.*, vol. 6, no. 7, pp. 822–830, 2012.
[16] Z. Su, X. Luo, Z. Deng, Y. Liang, and Z. Ji, "Edge-preserving texture suppression filter based on joint filtering schemes," *IEEE Trans. Multimedia*, vol. 15, no. 3, pp. 535–548, Apr. 2013.
[17] A. B. Watson, "DCT quantization matrices visually optimized for individual images," in *Proc. IS&T/SPIE's Symp. Electron. Imag., Sci. Technol.*, 1993, pp. 202–216.
[18] *BOWS-2 Dataset*. Accessed: Dec. 28, 2016. [Online]. Available: http://bows2.ec-lille.fr

MOHAMED HAMID received the B.Sc. (Hons.) degree in electrical and electronics engineering from the University of Khartoum and the M.A.Sc. degree in electrical and computer engineering from Concordia University, where he is currently pursuing the Ph.D. degree in electrical and computer engineering. His research interests include signal processing, biomedical signals, and neuroimaging.

CHUNYAN WANG received the B.Eng. degree in electronics from Shanghai Jiao Tong University, Shanghai, China, and the M.Eng. and Ph.D. degrees from Universite Paris-Sud, Paris, France.

She is currently a Professor of electrical and computer engineering with Concordia University, Montreal, QC, Canada. Her current research interests include digital image processing and VLSI circuits for signal processing.