

**DCT-Based Image Feature Extraction  
and Its Application in Image Self-Recovery and Image Watermarking**

**Mohamed Hamid**

**A Thesis  
in  
The Department  
of  
Electrical and Computer Engineering**

**Presented in Partial Fulfillment of the Requirements  
for the Degree of Master of Applied Science (Electrical and Computer Engineering) at  
Concordia University  
Montreal, Quebec, Canada**

**August, 2016**

**CONCORDIA UNIVERSITY  
SCHOOL OF GRADUATE STUDIES**

This is to certify that the thesis prepared

By: Mohamed Hamid

Entitled: “DCT-Based Image Feature Extraction and Its Application in Image Self-Recovery and Image Watermarking”

and submitted in partial fulfillment of the requirements for the degree of

**Master of Applied Science**

Complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____	Chair
Dr. R. Raut	
_____	Examiner, External
Dr. J. Y. Yu (CIISE)	To the Program
_____	Examiner
Dr. M. O. Ahmad	
_____	Supervisor
Dr. C. Wang	

Approved by: \_\_\_\_\_  
Dr. W. E. Lynch, Chair  
Department of Electrical and Computer Engineering

\_\_\_\_\_ 20

\_\_\_\_\_ Dr. Amir Asif, Dean  
Faculty of Engineering and Computer  
Science

# **ABSTRACT**

## **DCT-Based Image Feature Extraction and Its Application in Image Self-Recovery and Image Watermarking**

Mohamed Hamid

Feature extraction is a critical element in the design of image self-recovery and watermarking algorithms and its quality can have a big influence on the performance of these processes. The objective of the work presented in this thesis is to develop an effective methodology for feature extraction in the discrete cosine transform (DCT) domain and apply it in the design of adaptive image self-recovery and image watermarking algorithms.

The methodology is to use the most significant DCT coefficients that can be at any frequency range to detect and to classify gray level patterns. In this way, gray level variations with a wider range of spatial frequencies can be looked into without increasing computational complexity and the methodology is able to distinguish gray level patterns rather than the orientations of simple edges only as in many existing DCT-based methods.

The proposed image self-recovery algorithm uses the developed feature extraction methodology to detect and classify blocks that contain significant gray level variations. According to the profile of each block, the critical frequency components representing the specific gray level pattern of the block are chosen for encoding. The code lengths are made variable depending on the importance of these components in defining the block's features, which makes the encoding of critical

frequency components more precise, while keeping the total length of the reference code short. The proposed image self-recovery algorithm has resulted in remarkably shorter reference codes that are only  $\frac{1}{5}$  to  $\frac{3}{5}$  of those produced by existing methods, and consequently a superior visual quality in the embedded images. As the shorter codes contain the critical image information, the proposed algorithm has also achieved above average reconstruction quality for various tampering rates.

The proposed image watermarking algorithm is computationally simple and designed for the blind extraction of the watermark. The principle of the algorithm is to embed the watermark in the locations where image data alterations are the least visible. To this end, the properties of the HVS are used to identify the gray level image features of such locations. The characteristics of the frequency components representing these features are identifying by applying the DCT-based feature extraction methodology developed in this thesis. The strength with which the watermark is embedded is made adaptive to the local gray level characteristics. Simulation results have shown that the proposed watermarking algorithm results in significantly higher visual quality in the watermarked images than that of the reported methods with a difference in PSNR of about 2.7 dB, while the embedded watermark is highly robustness against JPEG compression even at low quality factors and to some other common image processes.

The good performance of the proposed image self-recovery and watermarking algorithms is an indication of the effectiveness of the developed feature extraction methodology. This methodology can be applied in a wide range of applications and it is suitable for any process where the DCT data is available.

# Acknowledgment

I would like to express my gratitude to my advisor, Dr. Chunyan Wang, for her excellent guidance and support during my studies. Her expertise and advice were integral to the completion of this research work and I am honored to have worked under her supervision.

I would also like to thank my parents for their unconditional love and support and my sister who is my biggest motivation to always give my best. I would also like to thank all my family and friends, whether near or far, for their support and care.

# Table of Contents

List of Symbols .....	ix
List of Abbreviations .....	xi
List of Figures .....	xii
List of Tables .....	xiii
Chapter 1 Introduction .....	1
1.1 Challenges, Motivation, and Objective.....	2
1.2 Scope and Organization of the thesis.....	4
Chapter 2 Background and Relevant Work .....	6
2.1 The 2-D DCT .....	7
2.2 Visual Models Based on the DCT .....	8
2.3 Feature Extraction in the DCT domain.....	9
2.4 Image Self-Recovery .....	11
2.5 Image Watermarking .....	13
2.6 Summary .....	16
Chapter 3 Feature Extraction in the DCT Domain .....	18
3.1 Correlation Between Texture Patterns and DCT Coefficients.....	19
3.2 Detection of Image Blocks That Contain Texture .....	24
3.3 Texture Classification in the DCT Domain .....	27
3.4 Summary .....	32

Chapter 4 Image Self-Recovery.....	34
4.1 Principles of the Algorithm.....	34
4.2 Feature Extraction.....	37
4.2.1 Texture Detection and Classification .....	37
4.2.2 Bit Assignment.....	39
4.3 Encoding .....	43
4.4 Authentication Bits .....	43
4.5 Embedding .....	44
4.6 Reconstruction .....	46
4.7 Simulation Results .....	46
4.7.1 Length of the generated Reference Code and Visual Quality of Embedded Images .....	47
4.7.2 Reconstruction Quality.....	48
4.8 Conclusion .....	57
Chapter 5 Image Watermarking.....	58
5.1 Principle of the algorithm .....	58
5.2 The Watermarking Algorithm and Its Implementation .....	60
5.2.1 Texture Detection and Selection of Image Blocks for Watermark Embedding .....	61
5.2.2 Detection of the Directions of Gray Level Variations .....	62

5.2.3	Selection of the DCT Coefficients for Embedding .....	62
5.2.4	Watermark Embedding .....	64
5.2.5	Watermark Extraction .....	66
5.3	Simulation Results .....	66
5.3.1	Visual Quality of the Watermarked Image .....	68
5.3.2	Robustness of the Watermark .....	71
5.4	Conclusion .....	74
Chapter 6	Conclusions and Future Work.....	75
6.1	Concluding Remarks.....	75
6.2	Suggestions for Future Work .....	78
References	.....	79



## List of Symbols

$\alpha$	Scaling factor used in watermark embedding.
$DC$	DCT coefficient with zero spatial frequency.
$H_n$	Authentication code of the $n$ th image block.
$K$	Parameter representing the strength of watermark embedding.
$K'$	Preset parameter used in watermark embedding.
$M$	Profile map.
$M'$	Profile map with the number of bits in each segment of the reference code included.
$M''$	Segmented profile map.
$M''_n$	The $n$ th segment of the segmented profile map.
$N$	Number of blocks in an image.
$Q$	Quantization step
$R$	Reference code generated from the DCT coefficients of the blocks of an image.
$R'$	Segmented reference code.
$R'_n$	The $n$ th segment of the segmented reference code.
$R''$	Reordered segmented reference code.
$R''_n$	The $n$ th segment of the reordered segmented reference code.
$S$	Number of bits in each segment of the reference code.
$x$	Original image.
$x'$	Watermarked image.
$x_n$	The $n$ th image block.

$X_n$	DCT matrix of the $n$ th image block.
$X(i, j)$	The DCT coefficient in the $i$ th row and $j$ th column.
$W$	The watermark.

# List of Abbreviations

2D-DCT	Two dimensional Discrete Cosine Transform.
CR	Correctness Rate.
DCT	Discrete Cosine Transform.
DWT	Discrete Wavelet Transform.
HVS	Human Visual System.
JND	Just Noticeable Distortion.
LSB	Least Significant Bit.
PSNR	Peak Signal to Noise Ratio.
RLF	Random Linear Fountain Coding.
SPIHT	Set Partitioning in Hierarchical Trees Coding.
SVD	Singular Value Decomposition.
TR	Tampering rate.
XOR	Exclusive OR logical operation.

# List of Figures

Fig. 2.1 Spatial frequency components of the 8x8 DCT.....	8
Fig. 2.2 Block of 8x8 in pixels partitioned into four sub-regions [1] .....	10
Fig. 4.1 Block diagram of an image self-recovery system with adaptive encoding .....	35
Fig. 4.2 Block diagram of the proposed image self-recovery algorithm .....	36
Fig. 4.3 The proposed feature extraction procedure .....	37
Fig. 4.4 The number of bits assigned to the DCT coefficients for the 13 texture profiles .....	42
Fig. 4.5 Comparison of the number of bits of the reference code .....	48
Fig. 4.6 The original test images.....	51
Fig. 4.7 Corrupted versions of the images of Fig. 4.6 .....	52
Fig. 4.8 Recovered versions of the images of Fig. 4.7.....	55
Fig. 4.9 The recovered regions in the cases of Fig. 4.8 (a) and (c).....	56
Fig. 4.10 Comparison of the average PSNR values of the recovered mages at different tampering rates.....	57
Fig. 5.1 Block diagram of the proposed Image Watermarking Algorithm.....	61
Fig. 5.2 DCT matrix in zigzag order with the coefficients groups corresponding to the directions of gray level variations shown in Table 1: (a) Vertical, horizontal and diagonal; (b) Horizontally dominant diagonal: and (c) vertically dominant diagonal [73].....	64
Fig. 5.3 Some of the test images and the used watermark.....	67
Fig. 5.4 Watermarked versions of the images in Fig. 5.3 .....	70
Fig. 5.5 The extracted watermarks from the watermaked version of Lena after several attacks.	72

# List of Tables

Table 3.1 The defined profiles .....	31
Table 3.2 Classification of blocks using the locations of the highest peaks in the DCT matrix ..	33
Table 4.1 Number of bits of the reference code generated using the proposed method.....	48
Table 4.2 Visual quality of the watermarked image .....	48
Table 4.3 The PSNR of the reconstructed images of Fig. 4.8 .....	50
Table 4.4 The reconstruction quality of the proposed method compared to that of [1] with a 50% Tampering Rate.....	50
Table 4.5 Comparison of the average PSNR values (in dB) of the reconststructed images at different tampering rates.....	50
Table 5.1 Detection conditions of the directions of gray level variations .....	63
Table 5.2 Watermark embedding.....	65
Table 5.3 Comparison of the PSNR of the watermarked images .....	71
Table 5.4 Comparison of CR at different JPEG quality factors .....	73

# Chapter 1 Introduction

Image processing has penetrated all aspects of modern life and become an important part in multimedia, communications, surveillance and security, medical imaging, and many more. While the widespread use of the internet and mobile devices facilitates the creation, transmission and sharing of digital images, it sparks an enormous concern about image security and hence creates a critical need for effective image protection techniques.

Image protection involves preservation of image ownership information, authentication of the image contents, detection of malicious alterations in image data, and restoration of the lost image information after attacks. Among the techniques for image protection, watermarking is usually used for image authentication and copyright protection among other applications. For image security, in particular, to enable images to withstand malicious tampering, one can embed some pertinent image information in the original image so that it can be used for image restoration. Such techniques are often referred to as image self-recovery or self-embedding techniques.

Although image processing techniques, such as the image protection techniques mentioned above, have different emphases in their development and target different applications, there are some common issues in their performance metrics, such as the simplicity of processing and the visual quality of the processed images. However, as image processing usually deals with a huge volume of pixel data representing diverse signal features, it usually requires a vast amount of computation. More optimized results and an efficient use of the available resources, in terms of information represented in image data and computation capacity, can be achieved by adapting the processing to local image features. To this end, it is critical to find which kind of local image

features are required for a particular application, how to extract them, and how to apply them in the design of image processing algorithms such as image watermarking and image self-recovery.

The research work presented in this thesis is in the topic domain of image feature extraction and its application in image protection processes. In the following sub-chapters, the challenging issues in this domain are addressed, the objectives of the work are identified, and its scope is specified. Also, the organization of the thesis is presented in the last section of the chapter.

## **1.1 Challenges, Motivation, and Objective**

Feature extraction can be performed in the spatial domain by various approaches, such as those based on high-pass filtering and/or local binary patterns [1-4]. The computations in such processes can be very complex especially if a wide range of variations need to be covered and classified. Feature extraction can also be carried out in the frequency domain. Image information is represented in a more compact manner in the frequency domain. It is known that most of visual information is concentrated in the lower frequency components and this property has been exploited extensively. However, one cannot extract all the significant image features from low frequency components only, and ignoring the signals in medium and high frequencies may risk the loss of important image information and potentially jeopardize the processing quality. Generally, including a wider frequency range in the process of feature extraction increases the required computation. Hence, in order to avoid excessive computations, the process is often designed to extract very limited "features", such as the direction of the most dominant edge of an image block or the levels of texture [5-12], instead of precise information about various local image patterns. Therefore, a new methodology to represent image features in the frequency domain should be developed by making good use of the compactness of information in the frequency components.

In image self-recovery, a reference code is generated from an image and embedded in the image itself to enable the regeneration of the lost or corrupted parts of the image if it was damaged. To achieve high reconstruction quality, longer reference codes are needed to represent the image more precisely. Embedding longer reference codes results in more alterations in the image data, which degrades the visual quality of the embedded image. Hence, a key issue in image self-recovery is to generate a reference code that represents the critical image information to achieve a good reconstruction quality while being short enough not to cause visible distortions in the embedded image. To accomplish this, the pertinent image features are needed so that the encoding can be made adaptive to the image patterns.

In image watermarking, a watermark, i.e. a logo or a secret key, is embedded into an image, and it should be extracted later to prove the ownership and/or the authenticity of the image. The watermark is required to be robust to sustain common image processes (e.g. filtering) and data transformations (e.g. image compression). Watermark embedding can be performed in the spatial domain or in the frequency domain. Spatial domain techniques tend to be simpler but frequency domain techniques generally yield more robust watermarks [13]. Increasing the robustness of the watermark usually makes larger data alterations in the original image, which results in more visible distortions in the watermarked image. The main challenge in image watermarking is addressing this inherent conflict between watermark robustness and the visual quality of the watermarked image. Adaptive watermarking techniques must be used to embed the watermark in locations where the image patterns would make it the least visible. To do so, the information about the image features is needed.

The objective of the work presented in this thesis is to develop an effective and computationally simple image feature extraction methodology and apply it in the development of



two adaptive algorithms for image self-recovery and image watermarking. The emphasis in the developing the image self-recovery algorithm is on using minimal data to represent the image information as accurately as possible in order to achieve good visual quality in the embedded and reconstructed images. The work in image watermarking aims at maximizing the watermark's robustness without causing visible distortions in the watermarked image. The effectiveness of the developed feature extraction methodology is critical in the design of both the image self-recovery and watermarking algorithms, and thus it can be evaluated through their performances.

## **1.2 Scope and Organization of the thesis**

To achieve the objective stated above, the research work in this thesis focuses on developing a feature extraction methodology to extract the image features in the frequency domain to make good use of the high information compactness. The discrete cosine transform (DCT) is one of the most extensively used transforms and it is a part of many processes. A feature extraction methodology in the DCT domain will be applicable in a wide range of applications, therefore the work in this thesis will target the DCT domain and an analysis of the representation of image features in the DCT domain will be conducted towards the development of the feature extraction methodology.

In the development of the image self-recovery algorithm, the image features will be used to achieve adaptive encoding by assigning appropriate numbers of bits to the DCT coefficients according to the gray level patterns. This way the code length will be variable with the image blocks prioritizing the DCT coefficients critical to the representation of each block. This approach is expected to lead to minimal length of the overall reference code and a good reconstruction quality. The work in image watermarking involves using the properties of the human visual system (HVS) to find the image patterns in which data alterations will be the least perceptible. The DCT-

based image feature extraction will be applied to identify these patterns. Once the best locations are identified, the embedding strength is then determined according to the local image signal so that the watermark robustness is maximized and the visual distortion is minimized.

The thesis will be organized as follows: In Chapter 2, the mathematical background of the DCT is presented along with the discussion of some relevant existing techniques of image feature extraction, image self-recovery and image watermarking. In Chapter 3, the proposed feature extraction methodology is described in detail. Chapter 4 is dedicated to the description and evaluation of the proposed image self-recovery algorithm. In Chapter 5, the proposed image watermarking algorithm is presented and evaluated. Chapter 6 is a summary of the contributions of this thesis.

## Chapter 2 Background and Relevant Work

Image feature extraction aims at finding the image characteristics that efficiently describe the image. Feature extraction is especially useful in adaptive image processes where image features are used to make adjustments in the process. As many image processes already include frequency transformations, extracting the required features from the frequency information of an image eliminates the need to add spatial domain techniques on top of the usually computationally extensive frequency transform. One of the most commonly used frequency transforms is the DCT which makes it very useful to be able to extract low level features directly from the DCT data of an image.

Feature extraction can be used to enhance the two image protection techniques to be discussed in this thesis which are image watermarking and image self-recovery, and the DCT is extensively used in both of them. In image watermarking, adaptive processing is needed in order to maximize the watermark's strength without causing visible distortions. In image self-recovery, adaptiveness helps reducing the number of bits used to encode image information while maintaining the reconstruction quality.

In this chapter, the background and some relevant reported techniques of feature extraction in the DCT domain, image watermarking and image self-recovery will be discussed. First an overview of the DCT is given in Subchapter 2.1 and a brief discussion of the visual models based on the DCT is presented in Subchapter 2.2, followed by background of feature extraction in the DCT domain, image watermarking and image self-recovery respectively in the following subchapters.

## 2.1 The 2-D DCT

In frequency domain, the image is decomposed into many frequency components with most of the image information being concentrated in a part of those components. This property of energy compaction along with the removal of redundancy are the main reasons why it is often preferred to process images in the frequency domain.

The DCT is one of the most popular frequency transformations in image and video processing due to its simplicity and high energy compaction. For an image block  $x$  of size  $N \times N$ , the type II 2-D DCT is defined as

$$X(k_1, k_2) = \frac{2}{N} u(k_1) u(k_2) \sum_{n_1=0}^{N-1} \sum_{n_2=0}^{N-1} x(n_1, n_2) \cos \frac{\pi(2n_1 + 1)k_1}{2N} \cos \frac{\pi(2n_2 + 1)k_2}{2N} \quad (2.1)$$

where  $k_1, k_2 = 0, 1, 2, \dots, N - 1$ ,  $u(0) = \frac{1}{\sqrt{2}}$  and  $u(k) = 1$  for  $n \neq 0$ .

The 2-D IDCT can be expressed as

$$x(n_1, n_2) = \frac{2}{N} \sum_{k_1=0}^{N-1} \sum_{k_2=0}^{N-1} u(k_1) u(k_2) X(k_1, k_2) \cos \frac{\pi(2n_1 + 1)k_1}{2N} \cos \frac{\pi(2n_2 + 1)k_2}{2N} \quad (2.2)$$

Each DCT coefficient represents a certain spatial frequency. Those patterns are sometimes referred to as basis functions. Fig. 2.1 shows the spatial frequency patterns of the 8x8 DCT which is the most common block size used in image processing and the size we will be used throughout this thesis. In the DCT domain, an image block is represented as a combination of these basis functions with different magnitudes and/or signs. To put it in different words, if the image blocks

are thought of as visual words, then the basis functions of the DCT are the visual alphabet that composes these words.

The first DCT coefficient,  $X(0,0)$ , is the *DC* coefficient. The *DC* coefficient has zero frequency in both the vertical and horizontal directions and it indicates the brightness of the image block since it corresponds to the average of the pixel values in the block. The remaining coefficients are called the *AC* coefficients. The *AC* coefficients closer to the *DC* coefficient have lower spatial frequencies and the frequencies increase as we move away from the *DC* coefficient in all directions. *AC* coefficients respond to gray level changes that are in the same direction as their spatial frequencies; for example, the coefficients in the first column respond to gray level changes in the vertical direction (horizontal edges) since their spatial frequencies are vertical.

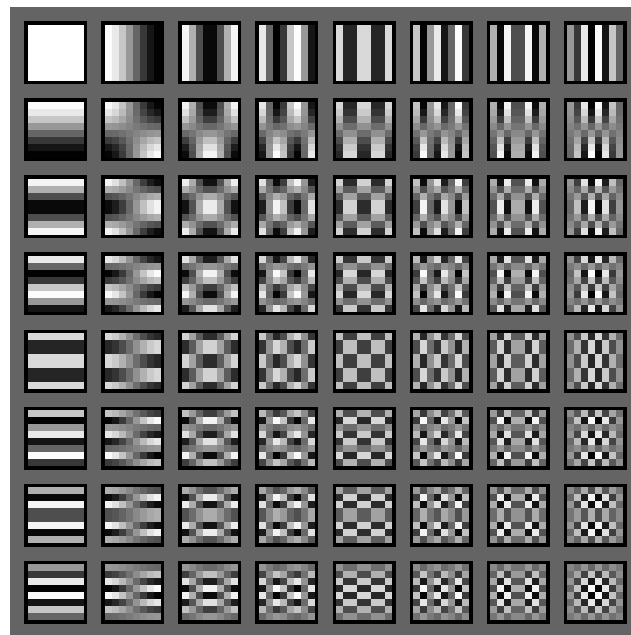


Fig. 2.1 Spatial frequency components of the 8x8 DCT

## 2.2 Visual Models Based on the DCT

The DCT is the most used transform in image compression because of its high energy compaction property. This sparked early studies of the visibility of the alterations made to DCT

coefficients in light of the properties of the human visual system (HVS) to determine suitable quantization matrices for use in image compression [14-20]. Andrew Watson used the properties of the HVS to develop a visual model for adaptive optimization of the DCT quantization matrices [21].

Following the same principle, the HVS properties were considered to define many DCT based Just Noticeable Distortion (JND) models which aim to define the thresholds beyond which modifications made to the DCT coefficients become visible [22-26]. These JND models are useful in a wide range of applications of image quantization and compression as well as in data hiding [27-31].

### **2.3 Feature Extraction in the DCT domain**

As mentioned previously, the DCT is commonly used in many image processing applications. This means that the DCT calculation is a part of many systems and if feature extraction was carried out using the DCT data directly, significant increase in the speed of the system can be achieved because of the elimination of extra spatial domain computations. Motivated by this, some attempts at image feature extraction directly from the DCT information of an image were made [5-11, 32].

Chang and Kang [5] derived a method to classify edge directions of single edges using the DCT coefficients directly from a modified version of the spatial domain technique in [12]. In the spatial domain technique, an image block is partitioned into four sub-regions as shown in Fig. 2.2, where  $S_{uv}$  is the average of the pixels in the corresponding partition. The values of  $S_{uv}$  are used to classify the edges into four directions:  $0$ ,  $\frac{\pi}{4}$ ,  $\frac{\pi}{2}$  and  $\frac{3\pi}{4}$ . The authors then derived equations to

calculate the values of  $S_{uv}$  directly from the DCT coefficients without the need to perform the IDCT first.

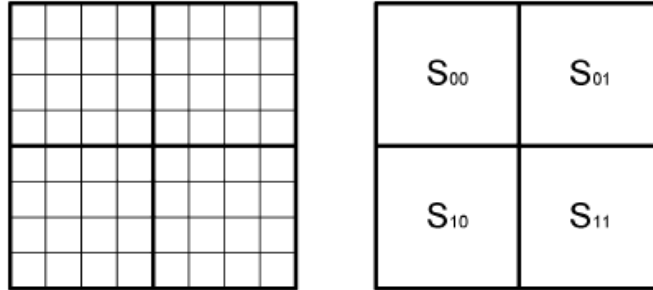


Fig. 2.2 Block of 8x8 in pixels partitioned into four sub-regions [5]

A hardware implementation based on this method was given by Vega-Pineda et al [6] and it was extended by Wang et al [7] to consider more directions. In those three methods [5-7], the focus is only on finding the orientations of single edges and no attention is paid to the possibility of more complex patterns. In addition, even though the mathematical computations needed in them is less than what is needed to calculate the IDCT, these methods still require a lot of computations.

Shen and Sethi [8] developed an algorithm capable of finding the orientations of single edges directly from the DCT information of the image. They presented an analysis of edge directions using comparisons of the two DCT coefficients with the lowest frequencies, namely  $X(0,1)$  and  $X(1,0)$ , and proposed using equation (2.1) to accurately obtain the edge orientation  $\theta$ . Although this method is fairly simple and DCT based, it also focuses only on the case of a single edge.

$$\tan \theta = \left( \sum_{v=1}^7 X(0, v) \right) / \left( \sum_{u=1}^7 X(u, 0) \right) \quad (2.1)$$

Li et al [9] followed a similar analysis to [8]. They used comparisons between 10 AC coefficients to determine the orientations of edges including the case of the presence of two parallel

edges. Eom and Choe [10] used comparisons between the amplitudes of DCT coefficients  $X(0,1)$  and  $X(1,0)$  to classify single edges into four orientations:  $0$ ,  $\frac{\pi}{4}$ ,  $\frac{\pi}{2}$  and  $\frac{3\pi}{4}$ . Jiang et al [11] included the coefficient  $X(1,1)$  in addition to  $X(0,1)$  and  $X(1,0)$  to achieve the same target. Park et al [32] used the averages of the coefficients in the first row, first column and the diagonal to do the same.

In all of the methods in [8-11], the use of a fixed set of coefficients to determine edge orientations can lead to inaccuracies since in some cases, the most significant DCT coefficients may not be inside the considered set. Moreover, all of the methods discussed in this subchapter, [5-11, 32], seem to focus on finding the orientations of simple edges only and ignore other possible patterns.

## **2.4 Image Self-Recovery**

The objective of image self-recovery, sometimes referred to as self-embedding, is to be able to reconstruct an approximate version of a corrupted part of an image using a code that was embedded in the image itself. The code is a compressed/lower quality version of the image itself and can be generated from the pixel domain image or from its frequency domain information. Many techniques have been reported for image self-recovery [33-48] and in this subchapter, the more pertinent ones will be discussed briefly.

Fridrich and Goljan [33] introduced the two basic encoding methods for image self-recovery that are followed by recent approaches. In the first method, a set of DCT coefficients is encoded in all image blocks. The generated reference code is then embedded into the least significant bits (LSB) of the pixel values of the original image. In the second method, the reference code is generated from the pixel values of the image. The gray level range of the image is reduced to form a low color depth version of the image which is then embedded in the original image.



In the algorithm proposed by Zhang et al [34], the gray level depth of the image is lowered by removing the 3 LSBs from each pixel, then the reference code is generated by encoding the 15 lowest frequency coefficient in each block of lower gray level depth image. The generated code is then embedded in the 3 LSBs of pixels of the original image. The reconstruction quality was later improved by employing compressive sensing in the reconstruction process in [35].

Korus et al [36] and Sarreshtedari et al [37] considered the communication of the reference code as an erasure channel problem. Korus et al [36] applied random linear fountain (RLF) coding to the code generated from the DCT information, which is generated in a similar way as in [34], before embedding. Sarreshtedari et al [37] used a set partitioning in hierarchical trees (SPIHT) source encoding algorithm to encode the multi-resolution wavelet transform coefficients.

In all of the aforementioned methods [33-37], all image blocks are encoded in the same way regardless of their characteristics. This resulted in large reference code which required the use of two or three LSB planes for embedding. This in turn would lead to a significant degradation of the visual quality of the image after embedding. Some approaches where the embedding process is made adaptive to the level of texture in image blocks were reported to reduce the length of the reference code in [38-40].

Korus et al [38] modified the method presented in [36] to include multiple quality profiles. They used the variance of the pixel values to classify image blocks according to their level of texture into low, medium and high texture blocks. The information about the level of texture is used to control the encoding process by targeting higher reconstruction quality for high texture. They managed to reduce the reference code length requiring 2 LSB planes instead of 3, however, the code is still considered long.

Qian and Feng [39] used inpainting to assist in the reconstruction process and reduce the length of the reference code. An edge operator is applied to each image block, then they are sorted according in descending order according to the number of edges they contain. The blocks are then classified to: high, medium and low texture and flat blocks, with the high texture being the highest 25% and the flat blocks being the lowest 25%. The flat blocks are excluded from encoding. The DCT is performed on the remaining blocks and their DCT coefficients are encoded with different code lengths according to their levels of texture. The generated reference code is embedded into the LSB of pixel values along with a map indicating texture levels of the blocks and authentication bits. In reconstruction, the blocks that were not included in the reference code because they were considered flat are recovered using the image inpainting method presented in [49]. This method yielded a shorter reference code requiring only 1 LSB plane for embedding but resulted in low reconstruction quality.

In the two discussed adaptive methods [38, 39], the embedding process is only made adaptive to the level of texture in image blocks and the gray level patterns of those blocks were not considered. This way a high number of bits is assigned to an image block to cover as many DCT coefficients as possible in order not to miss any image features represented by those coefficients. Whereas if the gray level patterns of those blocks were considered, some bits can be saved by ignoring the DCT coefficients that are irrelevant to the pattern and concentrating on the coefficients that are critical to represent the specific image block. Thus, achieving further reduction in the reference code length while maintaining good reconstruction quality.

## **2.5 Image Watermarking**

In image watermarking, a watermark is desired to be invisibly embedded into the original image. The watermark is generally required to be robust, i.e. to survive the application of common

image processes such as filtering and contrast adjustment and image data transformations such as JPEG compression.

Image watermarking algorithms can be divided into two main categories: spatial domain techniques [50-56] and frequency domain techniques [13, 57-72]. In spatial domain, watermark embedding is performed by directly modifying the pixel values. An example of such techniques is embedding the watermark in the LSB of the pixels as in the discussed image self-recovery algorithms. Other popular spatial embedding techniques are ones based on difference expansion [51-54] and embedding via histogram modifications [55, 56]. In frequency domain techniques, the embedding is carried out by modifying the frequency components of the image. The two most commonly used frequency transforms are the DCT [13, 57-66] and the discrete wavelet transform (DWT) [67-70]. Spatial domain techniques generally yield less robust watermarks than frequency domain techniques [13].

Increasing the robustness of the watermark means bigger alterations in the image data are required which leads to more visible distortions in the watermarked image. This conflict between robustness and invisibility is the main challenge in designing image watermarking systems.

Cox et al [13] introduced the idea of spread spectrum watermarking and viewed the frequency domain of an image as a communication channel. They argued that embedding the watermark in the most perceptually significant components of the image would increase its robustness, therefore, they embedded the watermark in the DCT coefficients with the highest magnitudes. While it is true that this makes the watermark harder to destroy, it undoubtedly degrades the visual quality of the watermarked image.

You et al [68] embedded the watermark in those singularities in the high frequency sub-bands of the DWT in order to improve the visual quality of the watermarked image as the HVS is less sensitive to alterations in high frequencies. Lee and Li [63] chose to embed the watermark in the lowest frequency components to increase robustness and embedded multiple duplicates of the watermark to increase resistance to cropping. Some approaches used a combination of singular value decomposition (SVD) with a frequency transform in order to balance robustness and invisibility of the watermark [73, 74] but that adds more computational complexity to the system.

These discussed approaches all accept some trade-off between the robustness and invisibility of the watermark. To maximize the robustness while maintaining the visual quality of the watermarked image, some adaptive approaches that made use of the HVS properties were reported [57, 61, 65, 70, 75].

Podilchuk and Zeng [57] used the work of Watson [21] to determine the upper bounds of watermark insertion in both the DCT domain and the DWT domain and adjust the embedding strength accordingly. In similar fashion, Zhi et al [61] explored the use of JND models in image watermarking. Wan et al [65] introduced a DCT based visual saliency model and used it to modulate the JND model used in [31] in order to avoid distorting the salient regions of the image.

In the adaptive methods discussed above, the embedding is made adaptive in terms of the embedding strength only without careful consideration of the locations of embedding within the image nor of the choice of DCT coefficients for embedding.

OuJun et al [62] proposed to choose the DCT coefficient used for embedding in each block according to the direction of texture in the specific image block. They divided the DCT coefficients into three groups representing the vertical, horizontal and diagonal directions. Then calculated the

total energy of each group and considered the direction of the group with maximum energy as the dominant direction. However, this approach does not necessarily produce the correct answer. For example, in a case where the DCT coefficient with the highest magnitude is outside the group that has the highest total energy, that coefficient might have a bigger influence on the direction variations than the entire group.

The work of Watson [21] specified three main HVS properties that should be exploited: The HVS' sensitivity decreases as the brightness increases, the HVS is less sensitive to changes in regions that contain high texture, and the dominant image pattern masks the visibility of other patterns of similar orientations. In each of the reported methods, only one or two of those properties are exploited and not all three. Moreover, in some of them, complex mathematical equations are needed to calculate the embedding parameters.

## **2.6 Summary**

In this chapter, some background on the DCT is presented and existing techniques in image feature extraction in the DCT domain, image self-recovery and image watermarking are discussed. The reported methods for image feature extraction in the DCT domain are only capable of determining the orientations of simple edges and are unable to detect other gray level patterns. In this thesis, a DCT based image feature extraction that is capable of distinguishing different gray level patterns is to be developed and it will be applied in the design of image watermarking and image self-recovery algorithms.

The existing methods for image self-recovery use long reference codes to represent the image and the adaptive ones only adapt to the level of texture in the image blocks. Significant reduction in the amount of data used to represent the image can be achieved if the encoding is

made adaptive to the gray level patterns by selecting only the DCT coefficients critical for the specific pattern only.

In image watermarking, most methods accept a trade-off between the watermark's robustness and invisibility and the adaptive methods attempting to solve the conflict do not fully exploit the HVS properties and do not yield optimal results. In this thesis, a DCT based image watermarking algorithm that adapts to the characteristics of the image and exploits the HVS properties in order to maximize the watermark's robustness without causing visible distortions will be developed.

## Chapter 3 Feature Extraction in the DCT Domain

The approaches to extract image features in the DCT domain discussed in Chapter 2 fall into two main categories. In the first category, the information used to extract the image features are obtained by means of a partial inverse DCT transformation [5-7], which requires less computation than the full inverse transform, but is still not computationally simple enough for many applications. The second category uses direct comparisons of certain DCT coefficient and/or comparison of some parameters calculated from a fixed set of DCT coefficients [8, 9, 11, 32], which is computationally simple. Nevertheless, in this category, only the lowest frequency coefficients are taken into consideration which leads to inaccuracies in the feature extraction process because the significant DCT coefficients for many image blocks may be outside the fixed frequency range examined. Furthermore, the reported approaches in both categories represent all the texture in the image as single edges and the final result is only the orientations of those simple edges.

The work in this thesis aims at developing an efficient methodology to analyze image features in the DCT domain and to apply this methodology to the design of image watermarking and image self-recovery algorithms. This methodology is to use a minimal amount of DCT data to effectively distinguish and represent as many different features as possible with a view of developing a computationally simple algorithm for feature extraction.

In Subchapter 3.1, the relationship of the DCT coefficients with the pixel domain image is analyzed. A method to distinguish image blocks with variations from flat blocks adaptively is presented in Subchapter 3.2. The classification of texture into different profiles is discussed in Subchapter 3.3 followed by a summary in Subchapter 3.4.

### 3.1 Correlation Between Texture Patterns and DCT Coefficients

As described in chapter 2, each DCT coefficient represents a spatial frequency component and the pattern of an image block is a combination of those spatial frequency components with different magnitudes and signs. The reported methods to extract the features directly from the DCT information tend to operate under the assumption that an image block can contain only a single edge and focus on determining the orientation of those edge using the lowest frequency coefficients. Even though blocks of  $8 \times 8$  pixels are small enough to contain only simple patterns of texture, it is possible for a simple pattern to consist of multiple edges in the same or different directions. Therefore, it is important to analyze the correlation between the different patterns and the DCT coefficients to identify the most critical elements that define more characters in different patterns rather than simple edge orientations only.

The DC coefficient of an image block represents the average of pixel gray levels in the image block. The AC coefficients respond to gray level changes in different directions, and so, their values and signs directly relate to the strength, shape and orientation of the texture in the image blocks. More importantly, it has been observed that the texture in an image block can be characterized by the most significant AC coefficients, referred to here as the DCT peaks, or simply the peaks.

These DCT peaks are the key elements and should be given primary importance in the analysis of the frequency components of an image block. To analyze the locations of these peaks in the DCT matrix, a number of images of  $512 \times 512$  pixels were used. Each of them is divided into 4096 blocks of  $8 \times 8$  pixels and the location of highest DCT peak of each block is recorded. Each of the graphs in Fig. 3.1 shows the spatial frequency distribution of the highest peak in the DCT matrices of a commonly used test image. The DCT coefficients are numbered in the zigzag order



shown in Fig. 3.1 (e). It can be observed that the peaks are concentrated in coefficients in the low to medium frequency range. Even though the majority of those peaks are in the lowest two frequencies, i.e. the coefficients numbered 1 and 2 in the zigzag order, a noteworthy number of blocks have their peaks in higher frequency coefficients representing features such as multiple edges. It is risky to neglect such blocks, especially since they may be in critical areas of the image and misrepresenting them can lead to a substantial loss of accuracy.

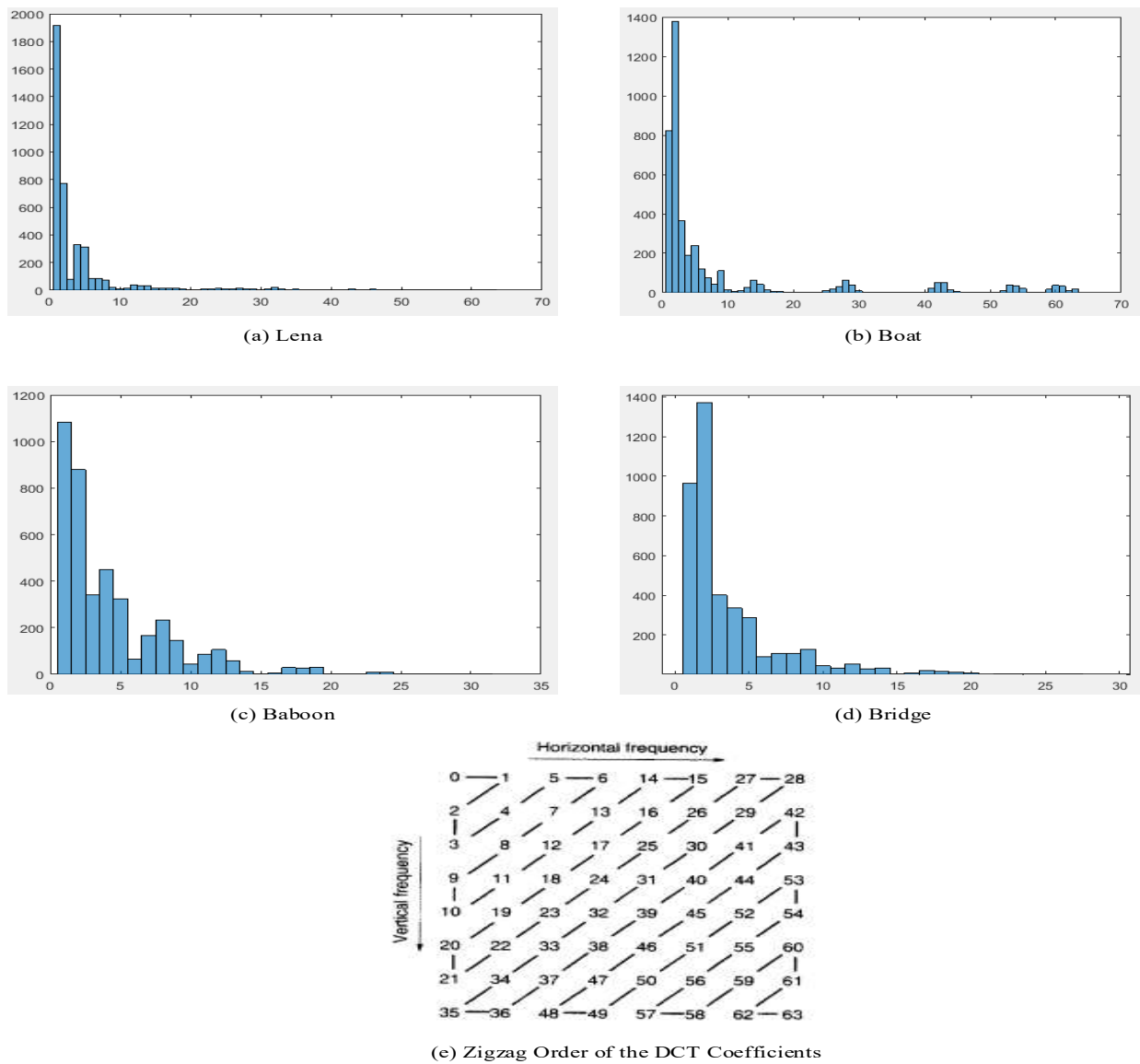


Fig. 3.1 Distribution of the highest peak in the DCT matrices extracted from different test images

The analysis in the following paragraphs will consider the identification of the most dominant orientations of edges in a block, the number of edges, the offset from the center; and the eventual presence of edges in different directions. A number of examples of image blocks extracted from real images are shown in Fig. 3.2 and they will be used throughout the analysis.

The most dominant edge orientation of the texture in an image block is reflected in the highest peak among the non-zero AC coefficients of the block. If an image block contains gray level changes in the horizontal direction (appearing as vertical edges), the highest peak in the DCT matrix is likely to be located in the first row. For example, in Fig. 3.2 (a) the pattern consists of a single vertical edge and the highest peak in the DCT matrix is the coefficient  $X(0,1)$ . In Fig. 3.2 (b), the image block contains two vertical edges and the highest peak is the coefficient  $X(0,2)$  which is also located in the first row of the DCT matrix. In contrast, the blocks in Fig. 3.2 (c) and (d), contain horizontal edges and the highest DCT peaks are found in the first column of the DCT matrix.

In addition, the location of the highest peak also indicates the number of parallel edges in the dominant orientation. If the block contains a single edge, the highest peak will be either  $X(0,1)$  or  $X(1,0)$ . On the other hand, if the block contains multiple parallel edges, the highest peak will be in coefficients of higher spatial frequencies. For example, the block in Fig. 3.2 (b), the image block contains two vertical edges so the highest DCT peak is  $X(0,2)$ . In Fig. 3.2 (d), the image block contains two horizontal edges, and the highest peak is  $X(2,0)$ . Whereas in the cases of the single edges in Fig. 3.2 (a) and (c), the highest peak is found at  $X(0,1)$  and  $X(0,2)$  respectively.

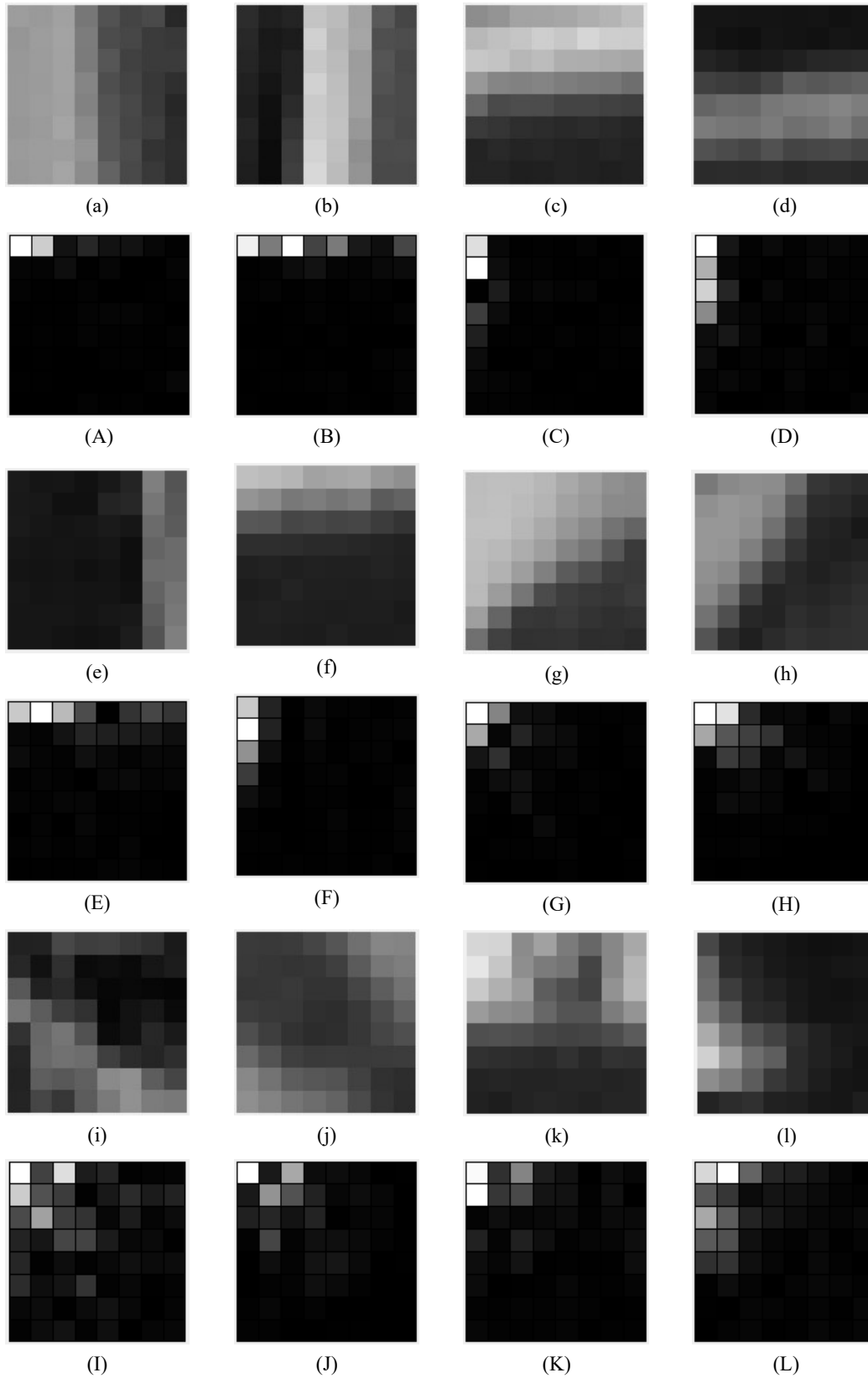


Fig. 3.2 Examples of 8x8 image blocks with simple patterns and their corresponding DCT matrices; (a)~(l) image blocks and (A)~(L) the corresponding DCT matrices respectively.

While the dominant orientation of edges and the number of parallel edges in that direction are indicated by the highest peak, more details are reflected by the second and third most significant DCT coefficients, referred to as the second and third peaks. If they are located in higher frequencies, they indicate offsets from the center and/or the presence of additional directions of change in the image block.

If the gray level pattern in an image block has an offset from the center, this offset can be indicated in the DCT matrix by the second and/or third highest peak being located in the same row or column as the highest peak. Such a case is illustrated in Fig. 3.2 (e), and the corresponding DCT matrix shows the highest peak to be at  $X(0,1)$  indicating a single vertical edge, and the second highest peak is at  $X(0,2)$  indicating an offset. In the same way, in Fig. 3.2 (f), both the highest peak and the second highest peak are located in the first column of the corresponding DCT matrix.

A single diagonal edge can be seen as the result of a gray level change in the vertical direction superimposed with a change of comparable amplitude in horizontal direction. Therefore, they are indicated in the DCT domain by significant values for both the vertical and horizontal coefficients with the lowest spatial frequencies, i.e.  $X(0,1)$  and  $X(1,0)$ . The highest peak will correspond to the more dominant direction and the second or third highest peak will be in the orthogonal direction. For example, in Fig. 3.2 (g), the edge is horizontally dominant, so the highest peak is found at  $X(1,0)$  and the second highest at  $X(0,1)$ . In Fig. 3.2 (h), the edge is vertically dominant, therefore, the highest peak is found at  $X(0,1)$  and the second highest peak is at  $X(1,0)$ .

In a more complicated case, some image blocks contain two different directions of gray level variations. In such blocks, the highest DCT peak will indicate the most noticeable direction of change. The second and/or third highest peaks will be located in higher frequencies in a different row or column to indicate the direction of the second grey level change. For example, the image

block shown in Fig. 3.2 (k) contains changes in both the vertical and horizontal directions and its DCT matrix is given as:

$$\begin{bmatrix}
 757.75 & 73.00 & \mathbf{130.76} & -44.73 & 28.00 & -1.17 & -21.42 & 11.84 \\
 \mathbf{386.86} & 81.84 & \mathbf{111.58} & -31.55 & 28.07 & -3.97 & -25.03 & 0.96 \\
 -6.99 & 22.24 & -11.76 & 13.74 & 18.46 & -5.32 & -17.66 & -15.48 \\
 -50.27 & -7.95 & -49.52 & 23.76 & 4.72 & -9.17 & -9.36 & -21.43 \\
 21.00 & -9.92 & -29.54 & 2.70 & -3.75 & 2.57 & 2.50 & -19.7 \\
 17.79 & 0.58 & -2.29 & -3.63 & -11.18 & 4.38 & 3.26 & -8.45 \\
 -5.76 & -3.69 & 6.34 & -5.18 & -9.00 & 5.45 & 0.26 & 2.56 \\
 -10.71 & -5.99 & 3.05 & -5.02 & -8.01 & -2.20 & 3.11 & -3.49
 \end{bmatrix}$$

The more noticeable change in Fig. 3.2 (k) is in the vertical direction, which is resembled by the strong horizontal edge. Thus the highest peak is at  $X(1,0)$ , while the second highest peak is at  $X(0,2)$  representing the smaller vertical change. Similarly, in Fig.3.1 (l), the vertical edge is the dominant one and it is indicated by the highest peak located at  $X(0,1)$  while the less significant parallel horizontal edges are indicated by the second highest peak at  $X(2,0)$ .

From this analysis, it is evident that the DCT peaks, found in the low to medium frequency range, are the critical elements to be analyzed to extract different characteristics from an image block. The highest peak indicates the dominant edge orientation as well as the number of parallel edges in that orientation, while the second and third highest peaks can indicate the offset of the gray level pattern and/or the presence of visible edges in a direction different from the dominant one.

### 3.2 Detection of Image Blocks That Contain Texture

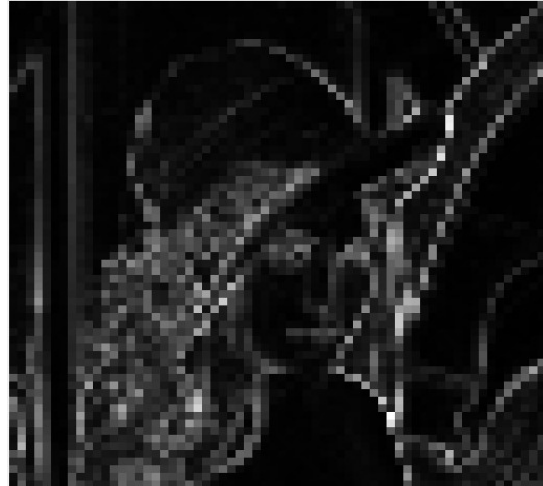
In the research work of this theses, identification of texture blocks, i.e. blocks that contain significant texture, is needed so that feature extraction can be applied exclusively to texture blocks rather than the entire image in order to reduce the overall computation of the processes of image

self-recovery and image watermarking. In the case of image self-recovery, it's logical to use a fewer number of bits to encode flat blocks and more bits for encoding blocks with texture in order to decrease the overall length of the generated code. In image watermarking, it is desired to avoid embedding watermarks in flat regions of an image because the HVS is sensitive to changes in these regions [21].

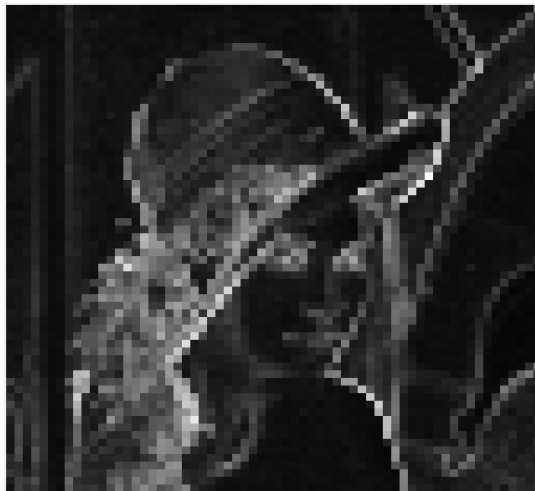
As discussed in the preceding subchapter, the presence of texture in an image block is indicated by the peaks in the DCT matrix. In flat blocks, we expect to see low values of the AC coefficients while in blocks that contain texture, at least one AC coefficient will have a significant value. As the HVS is less sensitive to texture in higher brightness regions of an image [21], the perception of texture is not only related to the magnitude of the AC components but also to the brightness of the image block observed. Therefore, it is important to include the brightness of the image block, which is indicated by the DC coefficient, in texture detection. In flat blocks, the ratio between the highest DCT peak and the DC coefficient should be low, while in texture blocks, it should be relatively high. Accordingly, the ratio between the highest DCT peak and the DC coefficient, *Peak/DC* ratio, provides a meaningful measure of the level of texture in the image block. Other parameters, such as the average of the magnitudes of the AC coefficients of the block or their variance [8], can also be used to measure the level of texture in the block. Fig. 3.3 shows *Peak/DC* ratios, AC coefficients average and variance for the test image "Lena". All the measures in Fig. 3.3 are normalized by dividing by their maximum value in the image.



(a) Original image



(b) *Peak/DC* Ratio



(c) Average of AC coefficients



(d) Variance of the AC coefficients

Fig. 3.3 Comparison of texture detection using peak/DC ratio and average and variance of the AC coefficients

It can be seen from Fig. 3.3 that all three measures offer indications of the levels of texture. The *Peak/DC* ratio appears to better illustrate different levels of texture than the other two, which can be attributed to the inclusion of the brightness of the block. Moreover, the computation of the *Peak/DC* ratio is simpler than the other two, and that makes it an attractive option for the purposes of the work presented in this thesis.

Texture blocks can be identified in an image by applying a threshold value to the *Peak/DC* ratios of the blocks. If the ratio for a block is higher than the threshold, it is considered to contain

texture; otherwise, it is considered a flat block. However, a more flexible approach would be to simply calculate the *Peak/DC* ratio for all image blocks and then selecting a portion of the blocks with the highest ratios, for example, selecting the highest 50% or 25% ratios. This approach allows the detection to be adjusted to the needs of the application. For example, if the application requires the selection of the most complex textures, such as in the case of an image watermarking algorithm that requires only a quarter of the image blocks for embedding the watermark. In such a situation, the blocks with the highest 25% of the *Peak/DC* ratios can be selected. Fig. 3.4 shows the results of texture detection using the *peak/DC* ratio when choosing the 50% and 25% of the blocks that have the highest *peak/DC* values.

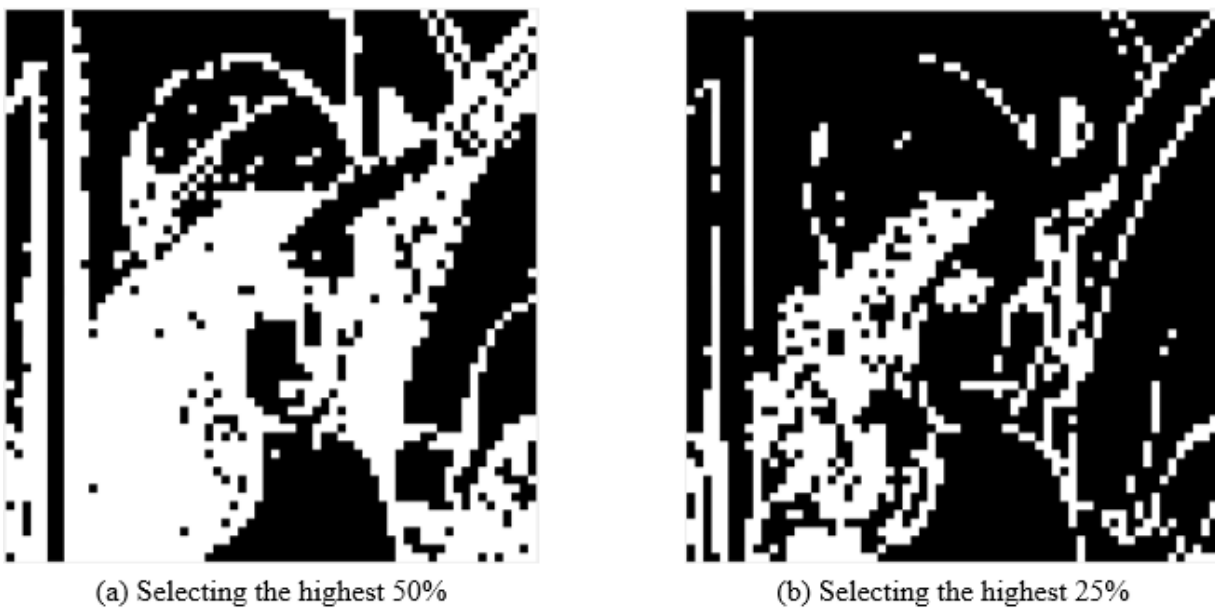


Fig. 3.4 Results of detecting texture using the *peak/DC* ratio

### 3.3 Texture Classification in the DCT Domain

Texture classification is applied to the identified texture blocks only and it is done using the DCT information. Since the DCT is usually included in many image processes, texture identification and classification add little computation to the process. In image watermarking, the



features of the original block are needed to determine the most appropriate locations for embedding the watermark with maximum strength while minimizing visible distortions. In image self-recovery, it helps in determining the most critical DCT coefficients that represent the block, which allows adaptive encoding of the image using minimal data while achieving good reconstruction quality.

The distributions of the highest peaks in the texture blocks of the four images used in Fig. 3.1 are shown in Fig. 3.5. Texture blocks were detected using the *Peak/DC* ratio and choosing the highest 50% of the blocks. The difference between Figs. 3.1 and 3.5 is that the former includes the flat blocks, in which the highest peaks are merely made by high frequency fluctuations. It can be seen from Fig. 3.5 that the peaks are concentrated in the coefficients numbered 1 to 20 in the zigzag order. It is thus reasonable to concentrate the analysis only on the 4x4 coefficients on the top left of the DCT matrix.

Among the 4x4 DCT coefficients, the two coefficients with the lowest spatial frequencies, i.e.  $X(0,1)$  and  $X(1,0)$ , are undoubtedly the most important, however, other coefficients are not negligible for good feature extraction. To simplify the inspection of the peaks, the remaining 13 AC coefficients taken into consideration are divided into 5 groups, in each of which the coefficients have a common nature in their spatial frequencies reflecting gray level variations in the same direction. For example, the coefficients in the first row of the DCT matrix other than  $X(0,1)$  indicate multiple parallel edges in the vertical direction, and similarly, the coefficients in the first column, except  $X(1,0)$  indicate multiple parallel edges in the horizontal direction. The DCT coefficients are divided into groups as shown in Fig. 3.6. The groups  $G_v$  and  $G_h$  represent parallel edges in the vertical and horizontal directions respectively;  $G_{d1}$  represent parallel diagonal edges;  $G_{d2}$  represents vertically dominant parallel diagonal edges; and  $G_{d3}$  represents

horizontally dominant parallel diagonal edges. The coefficients  $X(0,1)$  and  $X(1,0)$  together with the defined groups are used for texture classification.

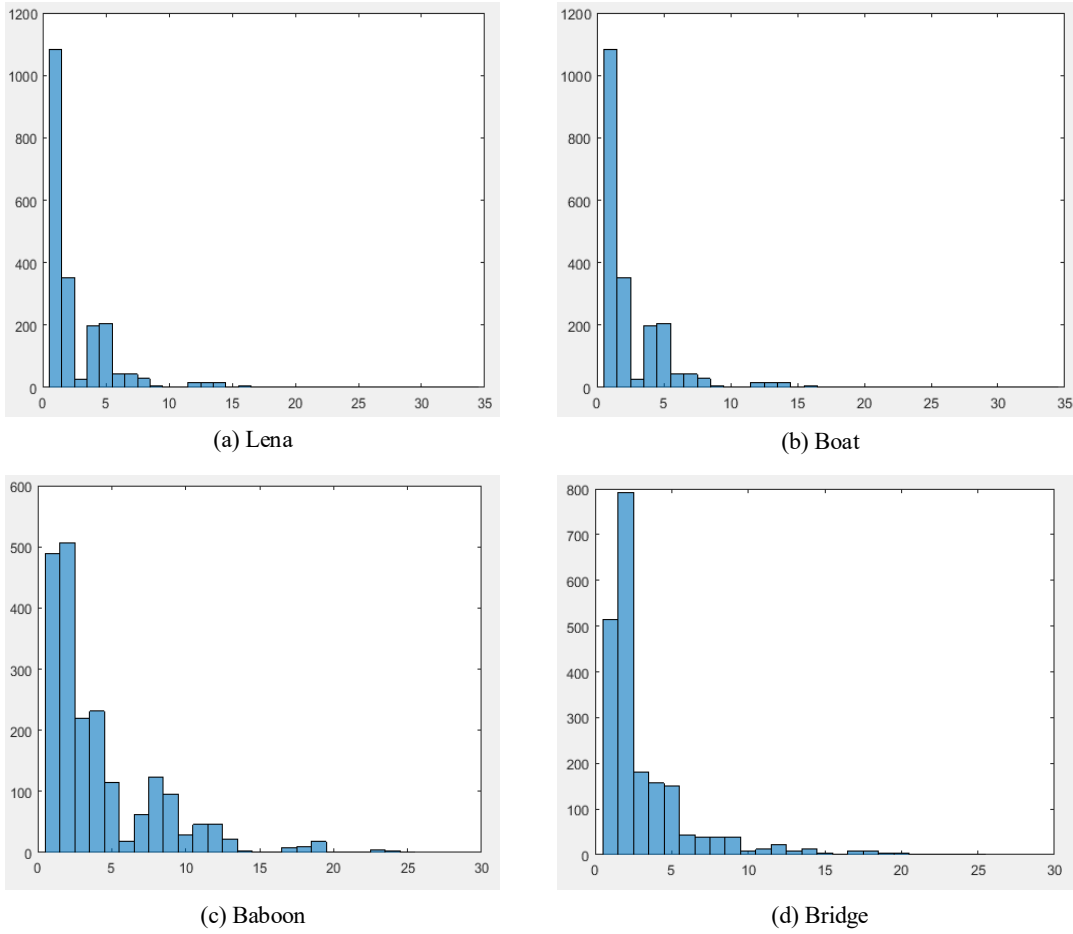


Fig. 3.5 Distribution of the highest peak in the DCT matrices extracted from high texture blocks

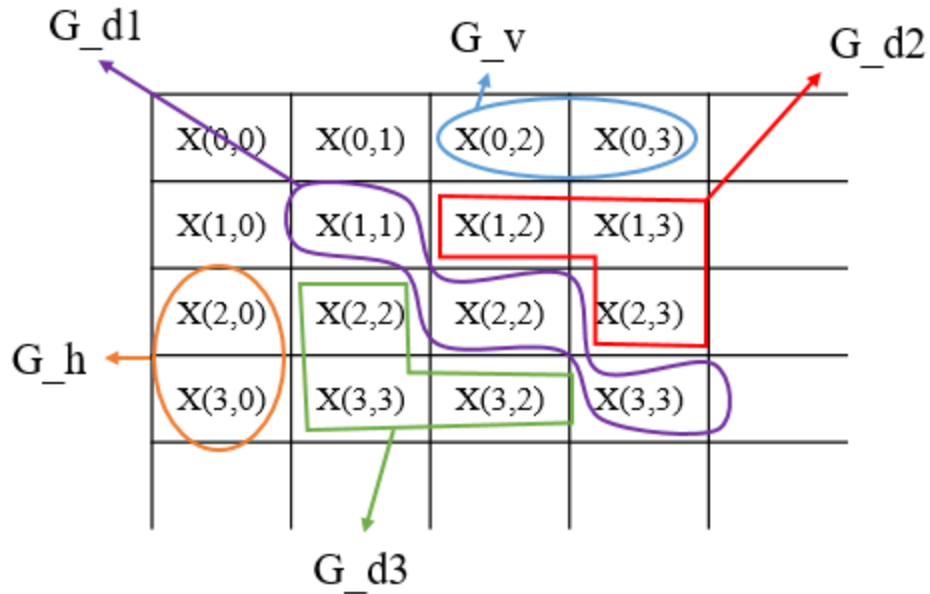















Fig. 3.6 DCT coefficients' groups in the low frequency 4x4 portion of the DCT matrix

As discussed previously, 8x8 blocks are small and contain only simple patterns of gray level variations. This enables us to define a small set of profiles that describes the possible patterns in 8x8 blocks. Thirteen different profiles are defined and they are shown in Table 3.1. Profiles H1, V1, D1 and D3 represent the cases of a single edge in different directions. The profiles, H2, V2, D2, D4 and D5 represent the cases of two or more parallel edges. The remaining profiles, namely T1, T2, Rec and Cross, represent the cases where more than one visible edge orientation exists in the block.

From the analysis in Subchapter 3.1, it is evident that each of the patterns presented in Table 3.1 can be represented by the highest three DCT peaks in the top left 4x4 coefficients of the DCT matrix. For example, if the highest peak is at  $X(0,2)$ , it indicates the presence of vertically dominant edges, which could be any of the profiles V2, D2 or Cross. The second peak helps distinguish between those three profiles. If it is located in first row, then the profile would be V2. For the profile D2, the second peak should be in group  $G\_d1$  or group  $G\_d2$ . In case of the profile

Cross, the texture pattern is characterized by a dominant medium frequency coefficient (in  $G_v$ ,  $G_h$ ,  $G_{d1}$ ,  $G_{d2}$  or  $G_{d3}$ ) subject to one of the other 2 peaks being in another medium frequency coefficient of an orthogonal spatial frequency.

Table 3.1 The defined profiles

Profile	Description	Examples
V1	Vertical edge	
V2	Vertical line (2 edges)	
H1	Horizontal edge	
H2	Horizontal line (2 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	

The classification conditions are summarized in Table 3.2, which covers all possible combinations of the locations of highest three peaks. For example, if the highest peak was  $X(0,1)$  and the second peak was  $X(1,0)$ , then regardless of the third peak, the block should have a single vertically diagonal edge and it will be classified as the profile D1.

Using this approach, only the locations of the highest three DCT coefficients in addition to the DC coefficient are needed to detect and classify texture blocks, and hence a minimal amount of DCT data is used. Additionally, the computation needed to locate the highest three peaks is reduced by concentrating on a  $4 \times 4$  portion rather than the full  $8 \times 8$  DCT matrix.

### **3.4 Summary**

In this chapter, the correlation between gray level variations and the DCT coefficients have been analyzed with the objective of developing a methodology to extract the characteristics of an image block directly from its DCT information. It has been found that gray level variations in the pixel domain can be effectively characterized by the highest DCT peaks. Those peaks are not only located in the lowest frequencies as indicated by many reported methods but can be found in the low to medium frequency range. In the developed methodology, the *Peak/DC* ratio is used to detect image blocks with texture then those blocks are classified to a small set of profiles based on the locations of the highest three peaks in the DCT matrix. The proposed method will lead to computationally simple feature extraction algorithms since it uses only the DC coefficient and the highest three DCT peaks, which makes it very suitable to be applied in image watermarking and image self-recovery.

Table 3.2 Classification of blocks using the locations of the highest peaks in the DCT matrix

1 <sup>st</sup> Peak	2 <sup>nd</sup> Peak	3 <sup>rd</sup> Peak	Profile
X(0,1)	G_v	G_h or G_d3	T1
		X(1,0)	D1
		otherwise	V1
	X(1,0)	×	D1
	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	G_v or G_d3	T2
		X(0,1)	D3
		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

## Chapter 4 Image Self-Recovery

In image self-recovery, a reference code is generated from the image, usually from its frequency information, and is embedded in the image itself. If that image is partially corrupted or damaged, the reference code will be extracted and used to recover the altered parts of the image. The reference code should be as short as possible to make computations simple and to minimize the visual distortions caused by embedding the code in the original image. However, it should be long enough to represent sufficient image features for an acceptable reconstruction quality.

In this chapter, an adaptive image self-recovery algorithm is proposed based on the feature extraction method presented in the preceding chapter. The objective is to design an image self-recovery system that represents the image using minimal data while achieving good reconstruction quality. The basic scheme of the algorithm is presented in Subchapter 4.1 and a detailed description of the proposed algorithm is given in the subchapters from 4.2 to 4.6. The simulation results are presented and discussed in Subchapter 4.7.

### 4.1 Principles of the Algorithm

Fig 4.1 shows a block diagram illustrating a basic scheme for image self-recovery with adaptive encoding [38-40]. It contains three major processes: classification of image blocks, encoding and embedding. The classification is carried out to group the blocks according to the complexity of their texture, and a particular encoding method is applied to the DCT coefficients of each block in the same group to generate the reference code, i.e. the code to be embedded and used to reconstruct the image blocks. Authentication bits, either preset by the user or calculated from the image information, are usually added to the code to enable the automatic detection of tampering.

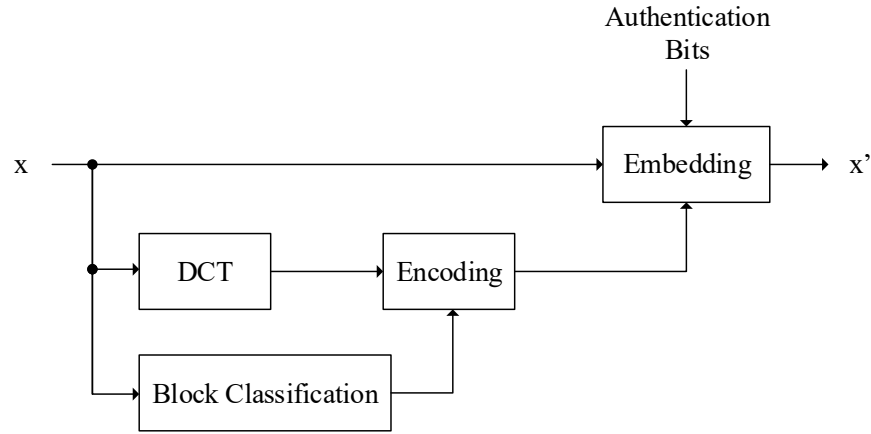


Fig. 4.1 Block diagram of an image self-recovery system with adaptive encoding

As discussed in Chapter 2, in existing image self-recovery methods, block classification is usually performed in spatial domain. The blocks are classified into a number of groups of different levels of texture without distinguishing patterns of textures in image blocks. For a particular group of a certain level of texture, all the blocks in that group are encoded in the same way and not discriminatively. As there can be different gray level patterns in a given level of texture, a relatively large number of bits are assigned to encode a wide range of frequency coefficients in order not to neglect significant features that may be represented in different frequency ranges.

To achieve the objective of minimizing the data volume and computations for image self-recovery, two issues are considered. The first is to classify the image blocks solely based on their frequency domain information, i.e. DCT coefficients, in order to spare the computations in spatial domain. It is known that edge information can be extracted from the DCT matrix [8] and the study presented in Chapter 3 demonstrated how the gray level variations can be characterized the peaks of the DCT coefficients. Hence the classification can be simply and effectively performed in the DCT domain.



The second issue is to adapt the encoding to the texture patterns of image blocks instead of simplistically adapting only to different levels of texture. Since the texture patterns in a block of 8x8 pixels can be classified to one of the profiles presented in Table 3.1 and each profile can be indicated by the highest DCT coefficients as shown in Table 3.2, one can encode the most critical DCT coefficients for the specific profile instead of those in the entire frequency range. In this manner, the image information encoding will be made more precise and adaptive to the image features of the profiles with a minimized code length. The block diagram of the proposed image self-recovery scheme is shown in Fig. 4.2, where feature extraction is the most critical process.

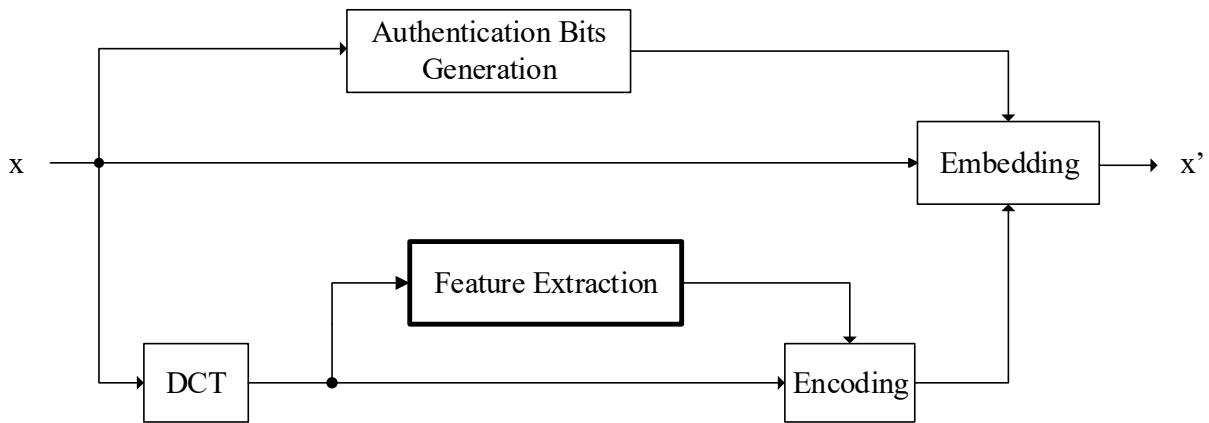


Fig. 4.2 Block diagram of the proposed image self-recovery algorithm

Feature extraction in the proposed method is composed of three steps: texture detection, texture classification, and bit assignment as shown in Fig. 4.3. Texture detection is a preliminary step to separate flat blocks, which are smooth and do not contain significant gray level variations, from blocks that contain texture. In texture classification, texture blocks are classified into a number of profiles according to their patterns. After the profile is identified, the most critical DCT coefficients are chosen for encoding with the number of bits assigned appropriately according to the profile.

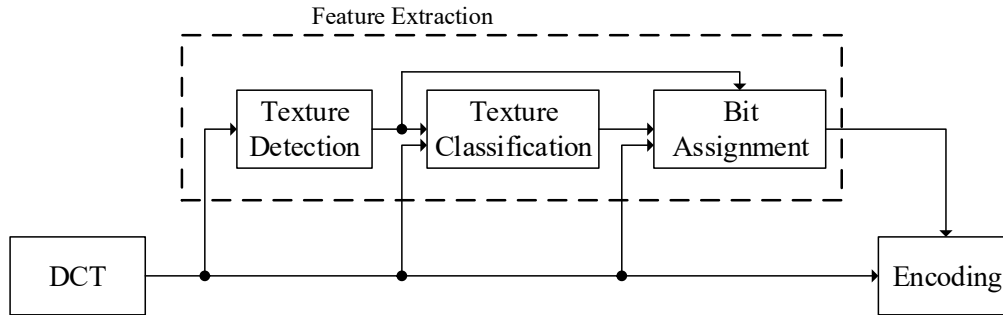


Fig. 4.3 The proposed feature extraction procedure

## 4.2 Feature Extraction

To efficiently use a small amount of data to represent the image, the length of code representing the image should be variable depending not only on the level of texture, but also on the texture profile. Thus, it is vital to be able to get a characteristic description of the image block through an appropriate block classification method. The feature extraction method developed in Chapter 3 is able to use minimal DCT data to distinguish different gray level patterns, which makes it suitable for application in image self-recovery. The application of this method in the proposed algorithm is described in detail in this subchapter.

### 4.2.1 Texture Detection and Classification

The first step of feature extraction is to separate flat blocks and texture blocks. Flat blocks do not contain significant details or meaningful information and require less attention in the feature extraction procedure. A flat block is considered an easy case for encoding and can be simply represented by its gray level average without visibly affecting the reconstruction quality. Texture blocks are the blocks that contain gray level variations representing meaningful information; therefore, they require more attention in analysis and it is important to encode the frequency components relevant to their patterns with the maximum possible accuracy.

In the proposed image self-recovery algorithm, the texture detection is performed using the *Peak/DC* ratios, as described in Chapter 3. The portion of blocks having the highest *Peak/DC* ratio values are considered texture blocks. For instance, one can choose 50% of blocks with the highest ratios in an image as texture blocks. The partitioning can be adjusted according to the requirements in different aspects. For example, if a higher reconstruction quality is required, a higher percentage of blocks should be selected as texture blocks. However, it must be kept in mind that higher accuracy comes at the expense of a longer reference code. To balance the reconstruction quality and code length, it is reasonable to select half of the image blocks as texture blocks as this way, it is more likely to capture most of the blocks with important details without assigning a big portion of the reference code to non-important blocks.

It is known that most of the visual information is concentrated in the low and medium frequency range, and in Chapter 3 it has been shown that for 8x8 blocks, it is sufficient to consider only the 4x4 DCT coefficients in the top left corner of the 8x8 DCT matrix to represent different image features because of the simplicity of the patterns contained by such small blocks. The 13 profiles defined in chapter 3 provide a very good representation of the possible gray level patterns that could appear in 8x8 blocks. They are defined following the analysis of different combinations of frequency components so that they offer a platform to identify the most critical frequency components needed to regenerate the pattern of each profile.

Texture classification is applied exclusively to the identified texture blocks. To classify a texture block, first, its DCT coefficients are examined and the highest three DCT peaks are located, then Table 3.2 is used to classify the texture block to one of the 13 profiles of Table 3.1. The results of texture classification make it possible to choose an appropriate set of coefficients to represent the particular gray level pattern in each texture block.

### 4.2.2 Bit Assignment

After the profile of an image block is identified, the most critical DCT coefficients to represent the block should be chosen and appropriate code lengths assigned to encode them with a view to minimize the overall number of bits. As the important DCT coefficients vary with gray level patterns, the DCT coefficients selected and the number of bits assigned to each DCT coefficient should vary with the texture profiles to adapt to the gray level patterns.

For a more precise representation of the image blocks, more bits should be assigned to the most critical coefficients that represent each profile. As revealed in the analysis presented in Chapter 3, the DCT peaks are the critical elements, therefore, they should be assigned more bits. Moreover, the texture profiles that appear more frequently should be encoded with more precision.

The DC coefficient reflects the gray level average of an image block. It is therefore the most important element in a DCT matrix regardless of the gray level pattern of the image block and it should be preserved with the maximum possible accuracy. From the DCT equation it can be seen that the DC coefficient is in fact equal to eight times the gray level average. This means that if the DC coefficient is quantized by 8, its maximum value would be 255 and 8 bits are needed to encode it without any loss.

As discussed previously, there are no significant gray level variations in flat blocks and their signals can be represented by their pixel average without any significant loss of accuracy. Hence, the DC coefficient is sufficient to represent flat blocks.

For the 13 texture profiles presented in Table 3.1, the encoding bits should be distributed appropriately among the important AC coefficients, shown in Table 3.2, while taking into consideration that some profiles appear more frequently than others. To be more specific, the

highest DCT peak is assigned the highest number of bits with respect to the other AC coefficients. Fig. 3.5 shows the distribution of the highest peaks and provides an insight on how frequently a profile can be seen in an image. It is reasonable that the highest peaks are assigned more bits in the cases of the most frequently appearing profiles. The number of bits assigned to the DCT coefficients for the 13 texture profiles is shown in Fig. 4.4.

The number of bits assigned to the locations of the highest peaks in different blocks varies from 8 to 5 depending on the profiles. The profiles H1 and V1 make up almost half of the texture blocks in an image, as can be seen in Fig. 3.5, and the locations of the peaks are determined, i.e.  $X(1,0)$  in case of H1 and  $X(0,1)$  in case of V1. Hence, 8 bits are assigned to those coefficients. In the case of the profiles D1 and D3, the amplitudes of the highest two peaks are comparable, and their locations are also determined; therefore, they are both assigned 7 bits. In the remaining profiles, the highest peak is less dominant than in H1 and V1. Additionally, its location is less determined. In other words, it can be found in a position belonging to one of the coefficient groups shown in Fig. 3.6. Each of the locations where the highest peak may appear is assigned 7 bits. The locations of the second and third peaks in the remaining profiles are assigned 4 to 6 bits according to their frequencies and the likelihood that a peak is located in them. Each of the profiles D2 and D4 can occur as a result of 4 different combinations of peaks, therefore they were divided into 4 sub-profiles as in Fig. 4.4 with two extra bits used to distinguish among the four possibilities. The profile Cross can also result from different combinations of peaks, however, as it is among the least encountered profiles, the assigned bits were spread among the AC coefficients instead of trying to represent each possible combination separately.

The DCT coefficients are quantized before encoding to reduce the range of their magnitudes and allow them to be encoded with acceptable accuracy with fewer bits. The quantization step,  $Q$ , used in the proposed algorithm is given by equation 4.1.

$$Q_{(i,j)} = \begin{cases} 8, & \text{if } i = j = 0 \\ 10, & \text{Otherwise} \end{cases} \quad (4.1)$$

As mentioned previously, the DC is quantized by 8 for it to be encoded by 8 bits without loss. The magnitude of the highest AC component was observed to rarely exceed 500. The maximum number of bits assigned to an AC coefficient is 7 bits which allows to encode values in the range  $[-63,+63]$ . Therefore, a quantization step of 10 is sensible since it would allow encoding values in the range  $[-630,+630]$ . If the value of a coefficient is too high to be encoded by the assigned number of bits, it is clipped to the maximum value allowed.

Since there are 13 possible texture profiles in addition to the flat profile, 4 bits are needed to indicate the profile of each block. Hence the length of the profile map would be  $4N$  bits. From Fig. 4.4, it can be seen that the image info in each texture block is represented by a code in the range between 29 and 60 bits in the cases of the profiles Rec and Cross respectively. However most of the texture blocks will be represented by 37 to 49 bits. A flat block is assigned only 8 bits since it is represented by its DC coefficient only. For an image consisting of  $N$  blocks, half of those blocks are considered flat blocks, thus, the contribution of flat blocks to the overall reference code is  $\frac{8N}{2}$ . For the texture blocks the maximum number of bits would be  $\frac{60N}{2}$  if all texture blocks belonged to the Cross profile while the minimum would be  $\frac{29N}{2}$  if all blocks belonged to the Rec profile. Therefore, the maximum possible overall length of the reference code is  $34N$  bits and it will always be between  $19N$  and  $34N$  bits.

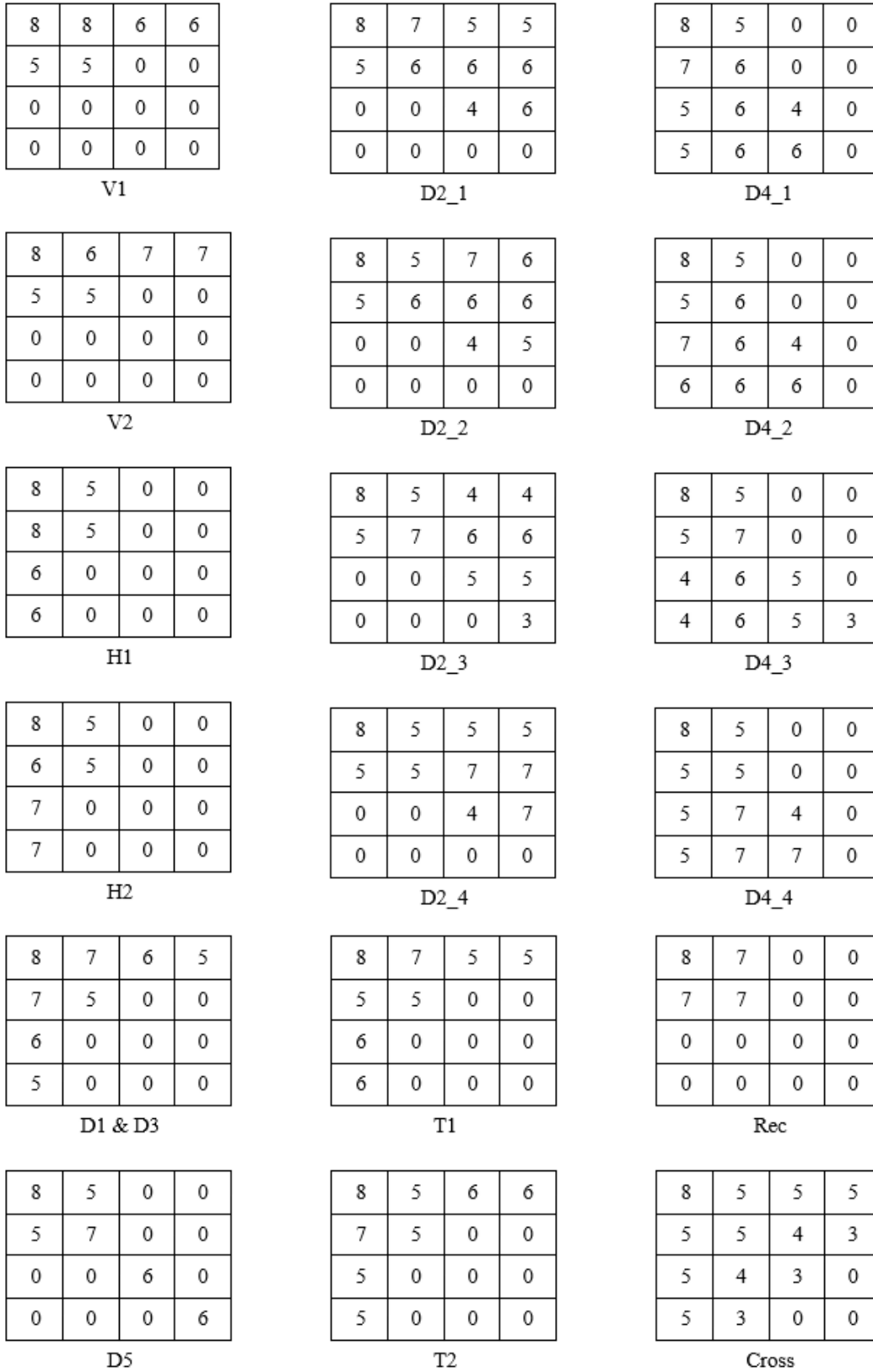


Fig. 4.4 The number of bits assigned to the DCT coefficients for the 13 texture profiles

### 4.3 Encoding

After bits are assigned to the DCT coefficients of an image block, its DCT coefficients are quantized and their values are converted to binary. The DC component is converted to an unsigned 8-bit integer and the AC coefficients are converted into signed integers with different lengths according to the bit assignment. In the cases of the profiles D2 and D4, two bits are added to the beginning of the code to distinguish between the four possible sub-profiles. The resulting binary integers are concatenated to form the block code. The coefficients are ordered column wise from top to bottom and left to right and the coefficients with 0 bits assigned are skipped. The image blocks are scanned in the same order and all the block codes are concatenated to form the reference code  $R$ .

The 4 bits indicating the profile of each block are recorded in the same order to generate the profile map  $M$ . Since this map is critical for decoding of reference code, it is important to duplicate  $M$  many times and embed it in different locations.

### 4.4 Authentication Bits

The inclusion of the authentication bits in the code is essential in order to automatically detect tampering and localize the tampered regions of the image at reconstruction. The authentication bits for each block are generated using the MD5 hashing method used in [36, 37, 39]. The MD5 hashing method generates a unique set of 128 bits for each set of data, which makes it very suitable for image authentication.

The authentication bits for each profile are generated from the gray level values of pixels. The MD5 hash code is calculated from all the pixel values with all LSBs set to 0 and the resulting 64 bits are divided into 16 groups. An XOR operation is performed on all the bits in each group to



produce  $H_n$ , the 16-bit authentication code for the image block, which will be embedded in the same block from which it was generated.

## 4.5 Embedding

The watermark to be embedded in the image consists of the generated reference code  $R$ , the profile map  $M$  and the authentication code  $H$ . It is required to be fragile, i.e. it should not survive tampering, in order to be able to use the authentication bits to detect any manipulations. In addition, the embedding method should permit embedding the entire set of codes in the cover image. To meet these two requirements, the watermark is embedded in the LSBs of the pixel values to produce the watermarked image. The embedding procedure is very similar to the procedure used in [39], however, because of the shorter reference code, not all pixels in the image are needed in embedding.

In general, an image consisting of  $N$  blocks of  $8 \times 8$  pixels has the embedding capacity of  $64N$  bits if the LSB of each pixel is used. As mentioned previously, the maximum possible length of the reference code  $R$  in the proposed algorithm is  $34N$ , which means that even with the addition of the profile map  $M$  and the authentication code  $H$ , it is sufficient to use only one LSB plane for embedding unlike the methods in [35-38], where 2 or 3 LSB planes are needed.

The following steps are performed for embedding:

1. Divide the reference  $R$  into  $N$  segments, where each segment contains  $S$  bits, to form  $R'$  where:

$$R' = \{R'_n: n = 1, 2, 3, \dots, N\} \quad (4.2)$$

$$S = \frac{\text{Length of } R}{N} \quad (4.3)$$

The index number  $n$  is mapped to  $m$  to reorder  $R'$  to  $R''$  using the following mapping function:

$$m = \begin{cases} (n+k) \bmod N, & \text{if } n+k \neq N \\ N, & \text{if } n+k = N \end{cases} \quad (4.4)$$

$$R'' = \{R''_n: n = 1,2,3, \dots, N\} \quad (4.5)$$

where  $k$  is a secret key.

As the length of  $R$  is determined by the features of the texture blocks and varies from one image to another, the value of  $S$ , calculated by equation (4.3), is important for decoding  $R$ . Since the length of  $R$  is between  $19N$  and  $34N$ ,  $S$  will be in the range between 19 and 34 bits. Hence  $S$  can be represented by  $S'$  where:

$$S' = S - 19 \quad (4.6)$$

The maximum value of  $S'$  is 15 which means only 4 bits are needed to encode  $S'$ .

2. Concatenate  $S'$  to the beginning and end of  $M$  and duplicate the result 3 times to form  $M'$ .

$$M' = \{S', M, S', S', M, S', S', M, S'\} \quad (4.7)$$

The duplication is done because this information is critical when decoding the reference.

$$\text{The length of } M' = 3 \times 4N + 3 \times 4 = 8N + 16 \text{ bits.} \quad (4.8)$$

$M'$  is divided into  $N$  segments where the first 16 segments contain 9-bits each and the remaining segments are 8-bits each. The result is

$$M'' = \{M''_n: n = 1,2,3, \dots, N\} \quad (4.9)$$

This way, the profile map will be embedded in multiple locations across the image and if some part of the image is corrupted, it can be retrieved from the uncorrupted parts.

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 watermarked block  $x'_n$ . Not all pixel values will be used because the maximum possible number of bits to be embedded will be  $9+16+34 = 59$ . The unused bits are left unchanged.

$$x'_n = embed(x_n, [H_n M_n'' R_n'']) \quad (4.10)$$

## 4.6 Reconstruction

First, the authentication bits  $H$  are extracted from the received image. The same procedure used for authentication is repeated to calculate the 16-bit authentication sequence for each received image block and they are compared to  $H$ . The blocks where a match was not found are considered corrupted. The profile map,  $M$ , and the number of bits in each reference code segment,  $S$ , are retrieved from the authentic parts of the image and they are used to decode the reference code. The reference code is then used to generate the DCT coefficients for the corrupted image blocks and the IDCT is performed to reconstruct those image blocks and replace the corrupted blocks.

## 4.7 Simulation Results

The proposed scheme was simulated using MATLAB to evaluate its performance in terms of the length of the generated reference code and the visual quality of the reconstructed image. The visual quality of an images is measured using the Peak Signal-to-Noise Ratio (PSNR). The images used in the simulations are commonly used gray-scale images of 512x512 pixels having different characteristics. The performance of the proposed algorithm is compared to the results of the methods reported in [35-39]. The method in [39] is an adaptive approach that shares the same objective of reducing the encoding data with the proposed method. The method in [38] is another adaptive method but with emphasis on controlling the reconstruction quality by providing high

quality recovery for the important contents at the expense of the less valuable background content. The remaining methods [35-37] are recent methods that show good reconstruction quality.

#### **4.7.1 Length of the generated Reference Code and Visual Quality of Embedded Images**

The lengths of the reference codes generated using the proposed method from different images in bits is shown in Table 4.1. Even though the code lengths in individual blocks are profile dependent, the total length varies only slightly from one image to another. The average reference code length is compared to those of the reported methods [35-39] in Fig. 4.5. From Fig. 4.5, it can be seen that the proposed method results in a reference code that is considerably shorter than the compared methods, requiring only between  $\frac{3}{5}$  to less than  $\frac{1}{5}$  of those of the methods reported in [35-39]. For the method reported in [37], the 2 LSB versions were used since they result in shorter codes.

The short code length facilitates embedding and leads to a very high visual quality in the watermarked image with an average PSNR of 51.64 dB as seen in Table 4.2. This because the proposed method uses only a portion of the LSBs of the pixels for embedding, whereas the compared methods require the 2 or 3 LSBs. According to a study presented in [37], the PSNR values of the watermarked image using 2 LSB planes [35, 37, 38] and the methods using 3LSB planes [35, 36] are 44.15 and 37.92 dB respectively. The decay in PSNR due to the use of an additional LSB plane is about 6 dB. The proposed method uses a reference code short enough to be embedded in only a portion of the LSBs of the pixels resulting in high visual quality. Moreover, it makes it possible to duplicate critical parts of the code to improve the chance of recovering severely damaged image blocks.

Table 4.1 Number of bits of the reference code generated using the proposed method

Image	Boat	Lena	Baboon	Peppers	Barbara	Bridge	Tiffany	Zelda	Pirate	Average
Ref. Code Length	114,041	114,477	119,441	115,992	116,991	115,862	115172	116,544	115,832	116,039

Table 4.2 Visual quality of the watermarked image

Image	Boat	Lena	Baboon	Peppers	Barbara	Bridge	Tiffany	Zelda	Pirate
PSNR (dB)	51.68	51.71	51.56	51.61	51.63	51.64	51.66	51.64	51.64

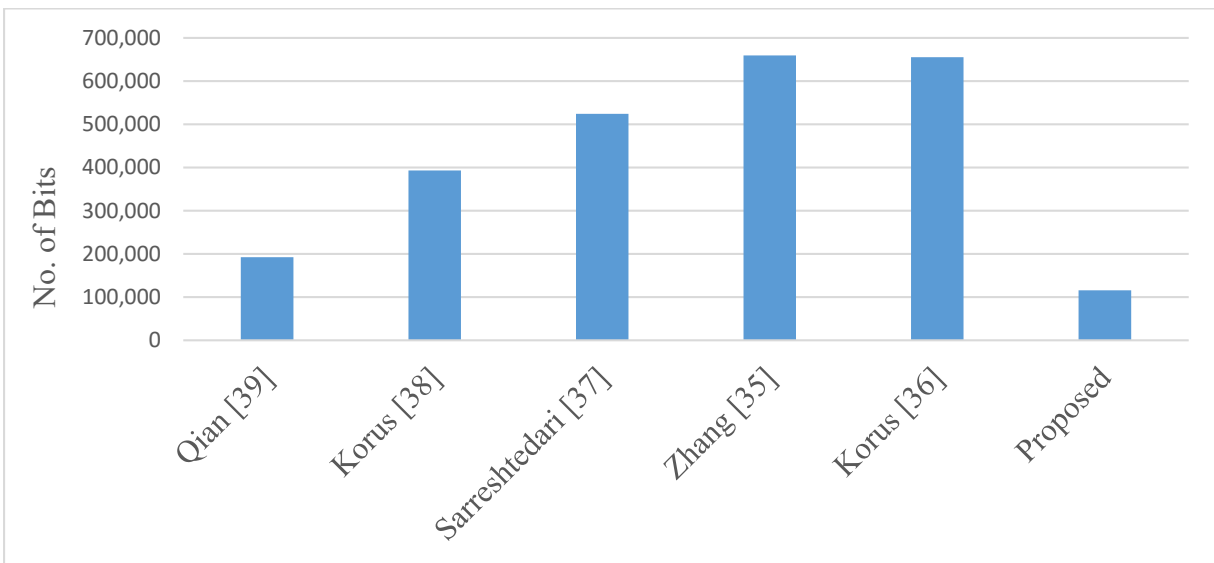


Fig. 4.5 Comparison of the number of bits of the reference code

#### 4.7.2 Reconstruction Quality

To evaluate the visual quality of reconstruction, parts of the test images, after encoding and embedding, are removed to produce the corrupted versions of the images, then they are recovered using the codes extracted from them. Examples of the original test images that were used in the tests and corrupted versions of them are shown in Fig. 4.6 and Fig. 4.7 respectively. The recovered versions are shown in Fig. 4.8, where it can be noticed that the corrupted regions were successfully recovered with an acceptable level of detail. Fig. 4.9 shows the recovered regions in the cases of Boat 1 and Lena 1 and offers a better view of the recovered area in comparison to its original

version. It can be seen from Fig. 4.9 that the general texture patterns of the objects were recovered with good quality. The recovered region in Fig. 4.9 (a) is clearly part of a boat and in (b) the recovered face is easily recognizable as the face of Lena. Table 4.3 shows the PSNR values of the recovered versions in Fig. 4.8. The PSNR values of the recovered versions are very good considering that the corrupted areas were all areas that contained significant textures.

Table 4.4 shows a comparison of the PSNR values of the recovered versions of various images using the proposed methods to the results presented in [39] with the tampering rate (TR) set to 50%, i.e. half of the blocks of each image are corrupted. The proposed scheme results in significantly higher PSNR values, about 2.5 dB higher in average, despite using a reference code that is almost half the length of the code used in [39].

The average reconstruction quality calculated from a set of 9 images at different tampering rates is compared to that of the methods in [35-38, 42]. The comparison is shown in Table 4.5 and Fig. 4.10. It can be seen from Fig. 4.10 that the proposed algorithm performs better than the methods in [35] and [42]-B. The methods in [42]-A and [37] seem to produce better results for a limited range of tampering rates, 20% and 30% respectively, while the proposed algorithm can successfully perform for tampering rates up to 50%. The methods in [36] has a constant reconstruction quality up to a TR of 50% and provides better results than the proposed algorithm for low TR values and the method in [38] provides better results for the entire TR range. However, it should be noted that the recovery quality is related to the information carried by the reference code, and the ideal case is to have maximum information given the shortest code. Among the cases presented in Table 4.5, the proposed method has a remarkably shorter code than all the others (70% and 82% shorter than those of [38] and [36] respectively) while its reconstruction quality is always above the average level at different tampering rates. Overall, the proposed feature extraction and

encoding methods lead to minimized reference code length with excellent density of image information in each bit of the code.

Table 4.3 The PSNR of the reconstructed images of Fig. 4.8

Image	Boat 1	Boat 2	Lena 1	Lena 2	Baboon	Peppers
<b>PSNR of the Recovered Version</b>	37.44	41.15	42.62	44.05	39.02	41.69

Table 4.4 The reconstruction quality of the proposed method compared to [1] with a TR of 50%

Image	PSNR of the Reconstructed Version (dB)	
	Qian [39]	Proposed
Peppers	31.8	34.85
Lena	32.8	34.27
Baboon	22.6	28.97
Zelda	36.8	37.45
Barbara	24.8	28.08
Tiffany	32.2	32.22
<b>Average</b>	<b>30.17</b>	<b>32.64</b>

Table 4.5 Comparison of the average PSNR values (in dB) of the reconstructed images at different tampering rates

Method	Tampering Rate				
	10%	20%	30%	40%	50%
[37]	40.5	40.5	40.5	0	0
[38]	44	44	39	37	34
[36]	36.4	36.4	36.4	36.4	36.4
[35]	37.2	34.8	29.1	28.1	27
[42]-A	40.7	40.7	0	0	0
[42]-B	31.7	31.7	28.7	28.7	25.8
Proposed	41.58	37.32	34.42	32.46	31.70



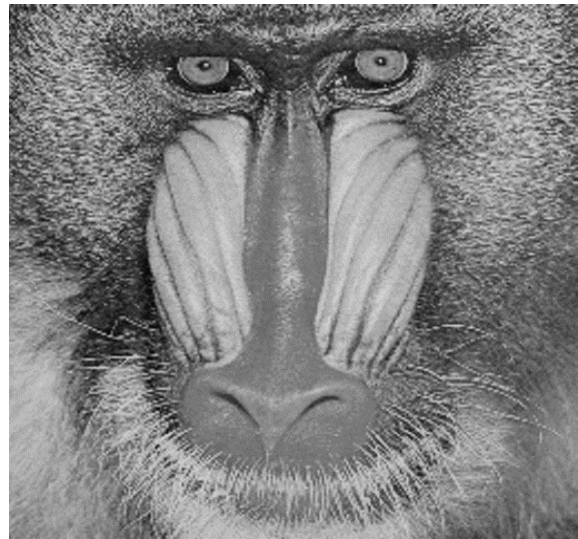
(a) Boat



(b) Lena



(c) Peppers



(d) Baboon

Fig. 4.6 The original test images





(a) Boat 1



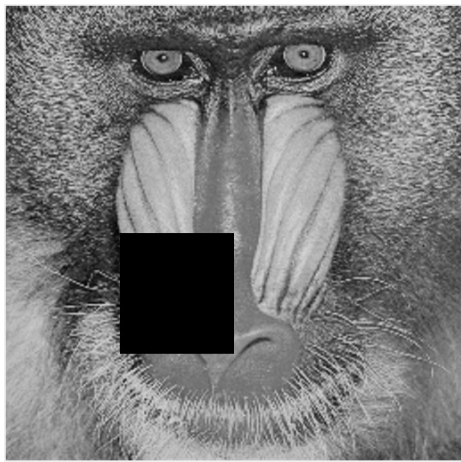
(b) Boat 2



(c) Lena 1



(d) Lena 2



(e) Baboon



(f) Pepper

Fig. 4.7 Corrupted versions of the images of Fig. 4.6



(a) Boat 1



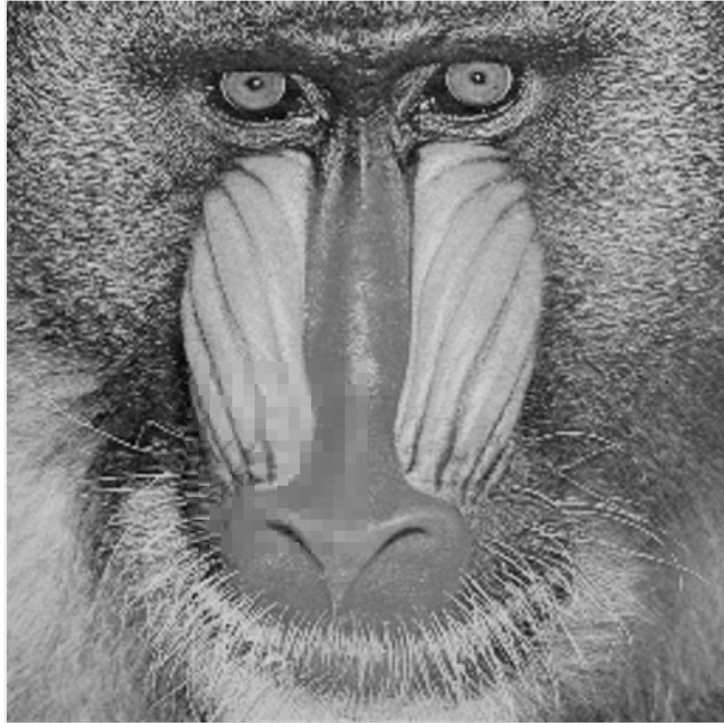
(b) Boat 2



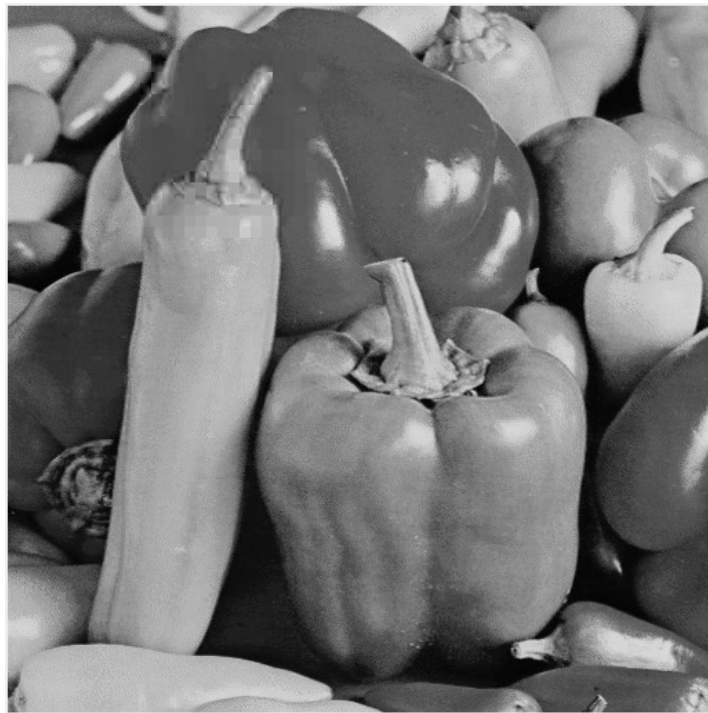
(c) Lena 1



(d) Lena 2

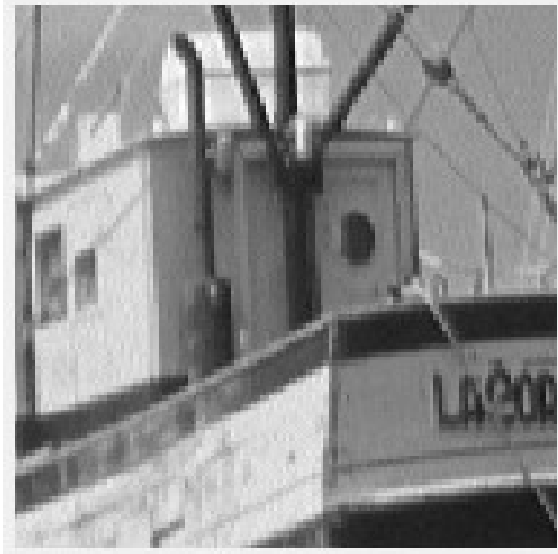


(e) Baboon



(f) Peppers

Fig. 4.8 Recovered versions of the images of Fig. 4.7

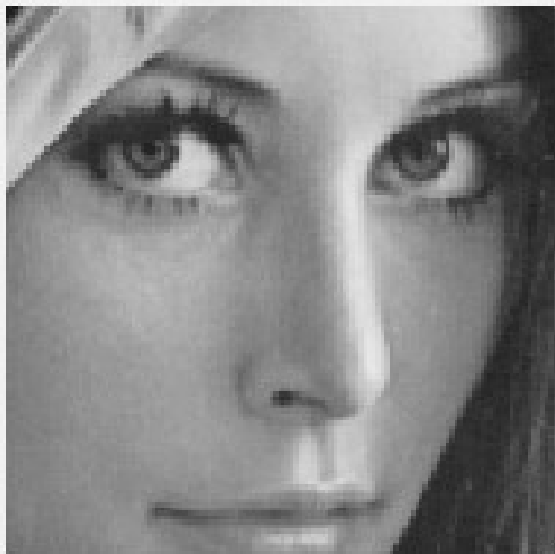


Original



Reconstructed

(a) Boat 1



Original



Reconstructed

(b) Lena 1

Fig. 4.9 The recovered regions in the cases of Fig. 4.8 (a) and (c)

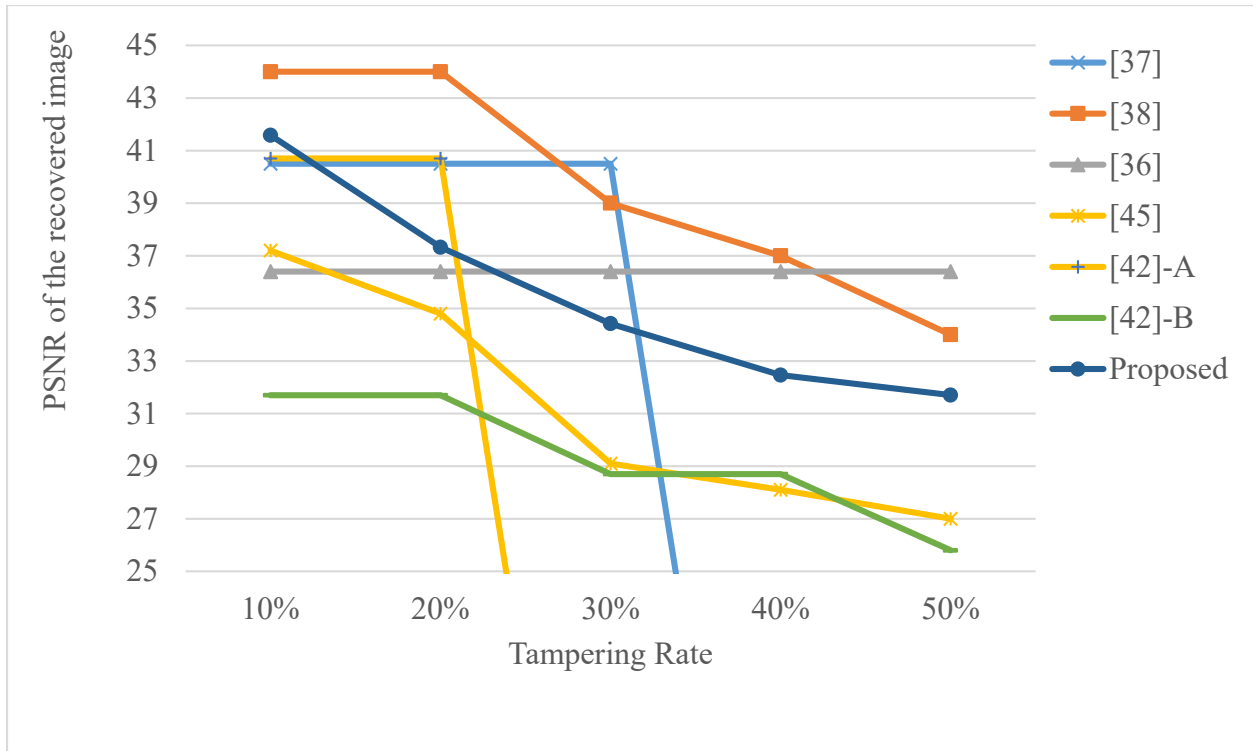


Fig. 4.10 Comparison of the average PSNR values of recovered mages at different tampering rates

## 4.8 Conclusion

In this chapter, the feature extraction method developed in the preceding chapter was applied to propose an adaptive image self-recovery algorithm. The proposed algorithm adapts to the gray level patterns rather than to the complexity of texture only. It is computationally simple and uses minimal data to encode the image information. The simulation results have shown that it yields good reconstruction quality and its performance is superior to recently reported methods in terms of achieving remarkable reductions in the encoding data.

## Chapter 5 Image Watermarking

In Image watermarking, it is desired to embed a robust watermark into an image without causing visible distortions in the watermarked image. As discussed in Chapter 2, the main challenge in image watermarking is addressing the conflict between robustness and invisibility. Some adaptive techniques aiming at resolving this conflict were reported [57, 58, 60, 61, 65, 66]. However, these methods focus mostly on finding the suitable regions of the image for watermark embedding and/or adjusting the strength of embedding and not on choosing the most suitable frequency components for embedding.

The objective of the work in this chapter is to develop a DCT based image watermarking algorithm that maximizes the embedding strength while minimizing visual distortions. The developed algorithm should be computationally simple and allow for the blind extraction of the watermark, i.e. without the presence of the original image. An image-adaptive approach is taken to achieve the objective and the feature extraction scheme presented in Chapter 3 is applied in the development of the algorithm with focus on determining the most suitable frequency components for embedding.

### 5.1 Principle of the algorithm

Image watermarking can generally be classified into spatial domain and frequency domain techniques. While spatial domain techniques are generally simpler, frequency domain techniques yield higher robustness [76]. As the DCT is a widely used and already a part of many image processes, embedding signals in the DCT coefficients gained popularity in image watermarking.

In DCT based watermarking schemes, the watermark data is embedded by modifying the DCT coefficients of the image. To increase the robustness of the watermark, greater alterations

have to be made in the DCT coefficients which leads to more visible distortions in the watermarked images. To solve this conflict between robustness and visual quality, the watermark must be embedded in the image regions of the image where data alterations are the least perceptible. To determine which regions of an image and which frequency components are the most suitable for embedding, it is necessary to consider the properties of the human visual system (HVS).

The work of Andrew Watson on developing visual models for adaptive quantization made use of the luminance and contrast masking properties of the HVS [21], which can be summarized in the following points:

1. The HVS is less sensitive to alterations in regions that contain significant gray level variations, i.e. regions that contain texture;
2. The presence of a dominant image pattern strongly masks smaller variations that have a similar orientation and spatial frequencies; and
3. The HVS sensitivity decreases as the background luminance increases, i.e. it is harder to perceive gray level variations in the brighter regions of an image.

According to the properties stated above: the watermark should be embedded in regions that contain sufficient gray level variations (GLV) to mask the alterations caused by embedding; the DCT coefficients used for embedding should be chosen in such a way that the resulting variations have the same orientations as the dominant original gray level patterns; and the embedding strength should be adjusted according to the brightness.

Accordingly, the image features needed for adaptive embedding are the levels of texture, gray level patterns and brightness of image blocks. If those features can be efficiently extracted from the DCT coefficients of the image, a fair amount of computations can be saved because there



will be no need for extra spatial domain processes. The feature extraction method presented in Chapter 3 is a computationally simple where the level of texture is measured using the *Peak/DC* ratio and texture patterns are classified using the highest DCT peaks. This makes the method presented in Chapter 3 very suitable for use in image watermarking.

In the proposed algorithm, the aforementioned HVS properties must be exploited to allow maximization of the embedding strength while maintaining high visual quality of the image. The watermark is to be embedded in the image blocks with significant gray level variations. Within such blocks, the DCT coefficients used in embedding will be chosen so that the alteration caused by embedding will have the same orientation as the dominant gray level pattern in the block, and the embedding strength will be adjusted according to the gray level mean of the block, i.e. the brightness of the block. In the following subchapter, a simple implementation of the discussed principles is presented.

## 5.2 The Watermarking Algorithm and Its Implementation

Fig. 5.1 shows a block diagram of the proposed algorithm. The input image  $x$  is divided into blocks of  $8 \times 8$  pixels which is the most commonly used size and the one used in Chapter 3. The watermark  $w$  is of  $m^2$  bits and it is to be embedded in the DCT matrices of  $m^2$  selected image blocks. After embedding, the watermarked image  $x'$  is obtained by performing the IDCT. A binary image is used as a watermark in the proposed algorithm to facilitate blind extraction.

As mentioned previously, the method presented in Chapter 3 will be used to extract the required features in the proposed algorithm. Feature extraction is composed of two steps: the first is to detect the texture level of the image blocks in order to choose the blocks with the most significant gray level variation for embedding. The second step is to detect the dominant

orientations of the gray level variations in those blocks so that appropriate DCT coefficients are chosen for embedding. It must be denoted that the orientations of gray level variations are generally perpendicular to edge orientations, for example, in a horizontal edge the gray level variations have a vertical orientation.

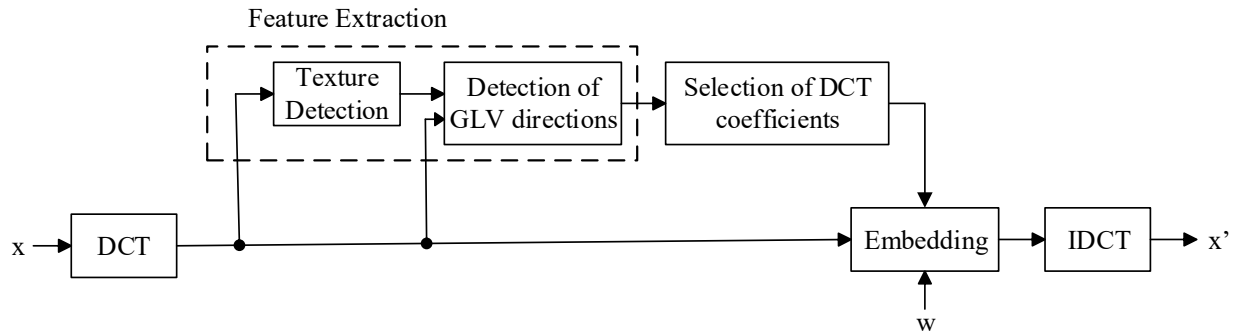


Fig. 5.1 Block diagram of the proposed Image Watermarking Algorithm

### 5.2.1 Texture Detection and Selection of Image Blocks for Watermark Embedding

As discussed previously, the image blocks with the most significant gray level variations must be chosen for embedding the watermark because the HVS is less sensitive to alterations in such blocks. The *Peak/DC* ratio is used, as described in Chapter 3, to detect the level of gray level variations in the image blocks. The image blocks with the higher ratios, referred to as texture blocks, are chosen for embedding. The threshold value of the *Peak/DC* ratio to separate texture and flat blocks is chosen according to the watermark size. For example, if a watermark of  $32 \times 32$  bits is used, the 1024 blocks with the higher *Peak/DC* ratios will be selected. This way the selection adapts to the image features and the watermark will always be embedded in the most suitable blocks in an image.

## 5.2.2 Detection of the Directions of Gray Level Variations

Knowledge of the directions of gray level variations is necessary to choose the most appropriate DCT coefficients for embedding the watermark data adaptively. In the method presented in Chapter 3, thirteen profiles are defined to describe the different gray level patterns in 8x8 image blocks. However, for the purposes of the work in this chapter, only the most dominant direction of gray level variations is needed while the number of edges and the secondary directions of change are less critical. Therefore, the texture classification explained in Chapter 3 can be simplified for use in the proposed watermarking algorithm. For example, the dominant direction of variations is the same for the profiles, V1, V2 and T1, i.e. the horizontal direction, therefore in the proposed algorithm, there is no need to distinguish among them.

In the proposed algorithm, five directions of gray level variations are considered: vertical, horizontal, diagonal, vertically dominant diagonal and horizontally dominant diagonal. Following the same reasoning as in Chapter 3, the dominant direction of variations can be detected using the locations of the highest two DCT peaks in the DCT matrix as shown in Table 5.1. The detection conditions in Table 5.1 are evaluated sequentially starting from the top. For example, in case of a block with horizontally dominant diagonal variations, the first three conditions will be found false and the fourth will be true.

## 5.2.3 Selection of the DCT Coefficients for Embedding

In the proposed algorithm, 1 bit of the binary watermark is embedded into each of the selected image blocks, and a single DCT coefficient must be chosen for embedding in each block. As mentioned previously, the chosen coefficient must have a spatial frequency that has the same orientation as the gray level variations in the original block. This way, the distortions resulting from embedding will be masked by the original variations. However, alterations to the highest

DCT peak must be avoided in the embedding process since it reflects the most dominant orientation as established in Chapter 3. In Addition, the chosen coefficient must be of a medium frequency, because alterations in the low frequency components are easier to perceive in the spatial domain and data embedded in high frequencies can be easily removed by simple low pass filtering without degrading the visual quality of the image.

Table 5.1 Detection conditions of the directions of gray level variations

<b>Direction of Gray Level Variations</b>	<b>Example</b>	<b>Detection Condition</b>
Horizontal	Profiles V1, V2 and T1	Both peaks are in the first row.
Vertical	Profiles H1, H2 and T2	Both peaks are in the first column.
Diagonal	Profile D5	The highest peak is on the diagonal line.
Horizontally Dominant Diagonal	Profiles D1 and D2	The highest peak is above the diagonal line.
Vertically Dominant Diagonal	Profiles D3 and D4	The highest peak is below the diagonal line.

To facilitate the selection of coefficients and embedding, the DCT coefficients are divided into 5 groups corresponding to the five directions shown in Table 5.1, where the coefficients in each group have spatial frequencies of similar orientations to the corresponding direction. The groups are shown in Fig. 5.2. After detecting the direction one coefficient from the corresponding group will be chosen for embedding the watermark bit.

In Fig. 5.2, the locations of the coefficient to be chosen for embedding in each case are shown. The coefficient indicated by a solid circle is chosen if it is not the highest DCT peak, otherwise, the coefficient indicated by the dashed circle will be chosen. For example, if the gray

level variations were detected to be in the vertical direction, the coefficient  $X(4,0)$  will be chosen, unless it is the highest peak, in which case, the coefficient  $X(3,0)$  will be selected.

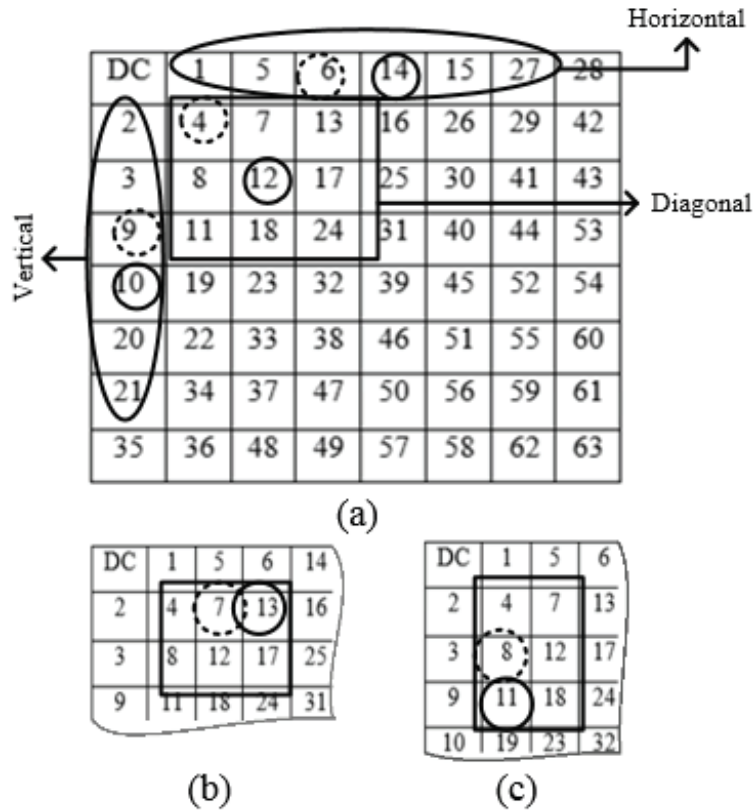


Fig. 5.2 DCT matrix in zigzag order with the coefficients groups corresponding to the directions of gray level variations shown in Table 1: (a) Vertical, horizontal and diagonal; (b) Horizontally dominant diagonal: and (c) vertically dominant diagonal [77]

### 5.2.4 Watermark Embedding

The watermark embedding process should be reversible and allow for blind extraction of the watermark. To facilitate blind extraction, the embedding should be carried out by modifying some relationship between the chosen coefficient and other DCT coefficients so that the extraction can be simply performed by testing this relationship without need for the original image. It should also be noted that the sign of the chosen coefficient should not be modified.

As the HVS is less sensitive to image data alterations in bright regions, the strength of embedding can be increased for blocks with higher gray level means. The gray level mean can be extracted from the DC coefficient by simply dividing it by 8 as can be seen from the DCT equation.

In the proposed algorithm, the watermark is embedded by making the amplitude of the chosen coefficient greater or less than the average of the amplitudes of the remaining coefficients in its direction group. The watermark is embedded as shown in Table 5.2, where:  $w$  is the watermark bit,  $C$  is the original value of the chosen coefficient,  $C'$  is the new value of the chosen coefficient, and  $AVG$  is the average of the amplitudes of the coefficients in the coefficients' group representing the detected direction of variations, calculated excluding the chosen coefficient and the highest DCT peak if it happened to belong to the group.

Table 5.2 Watermark embedding

$w = 0$	$w = 1$
$ C'  = \begin{cases} AVG - K & ,  C  > AVG - K \\  C  & , Otherwise \end{cases}$	$ C'  = \begin{cases} AVG + K & ,  C  < AVG + K \\  C  & , Otherwise \end{cases}$

The value of  $K$  determines the strength of watermarking and it is controlled by the value of the DC coefficient as shown in Equation 5.1:

$$K = K' + \alpha \cdot \frac{DC}{8} \quad (5.1)$$

where:  $DC$  is the value of the DC coefficient,  $K'$  is a predetermined constant, and  $\alpha$  is a scaling factor.

### 5.2.5 Watermark Extraction

In the proposed algorithm, the watermark can be extracted without the presence of the original watermark. The received image is divided into 8x8 blocks, then blocks with the higher *Peak/DC* ratios are considered the ones that contain the watermark bits. The direction of gray level variations in each of those blocks is then determined, the DCT coefficient used for embedding is identified, and the average of the amplitudes of the coefficient group representing the determined direction is calculated in the same way described for the embedding process. Then the watermark bit is extracted from each block using the following equation:

$$w' = \begin{cases} 0, & C \leq AVG \\ 1, & C > AVG \end{cases} \quad (5.1)$$

where:  $w'$  is the extracted watermark bit. The binary image formed by the extracted bits is the extracted watermark.

### 5.3 Simulation Results

The proposed algorithm is evaluated in terms of the visual quality of the watermarked image and the robustness of the watermark against several attacks. The PSNR is used to assess the visual quality of the watermarked image while the correctness rate (CR) is used to evaluate robustness. The CR is simply the ratio of the correct bits to the total number of bits in the extracted watermark. Preliminary results of the proposed algorithm are presented in [77].

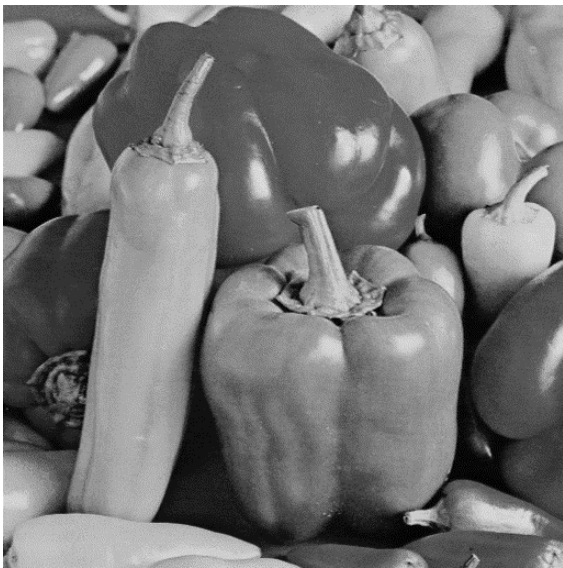
In the simulations, a binary watermark of 32×32 bits has been embedded into several commonly used gray level test images of 512×512 with different characteristics were used. Fig. 5.3 shows some of the images and the watermark. The parameters  $K'$  and  $\alpha$  have been preset to 10 and 0.05 respectively in the simulation.



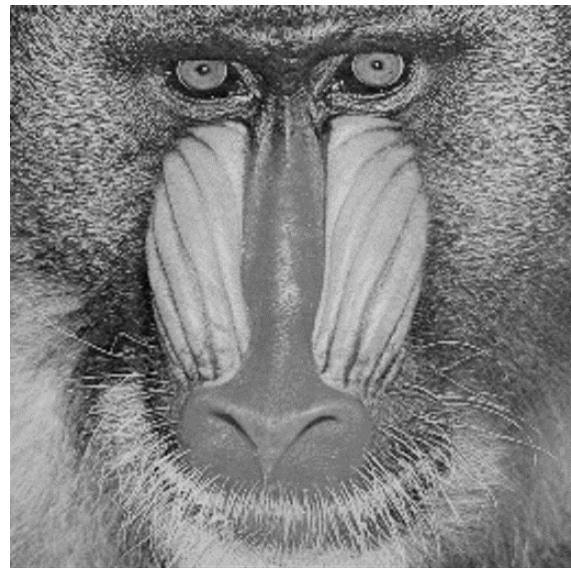
(a) Boat



(b) Lena



(c) Peppers



(d) Baboon



(e) The Watermark

Fig. 5.3 Some of the test images and the used watermark



The performance of the proposed algorithm is compared to the those of the algorithms in [61, 63, 65, 66, 68], all of which use binary watermarks and are designed for blind extraction. In [61], four JND models were applied to the development of the watermarking methods. We will compare to the two methods based on the JND models in [22] and [25] because they yielded better results. They will be denoted by Zhi-1 and Zhi-2 respectively.

### 5.3.1 Visual Quality of the Watermarked Image

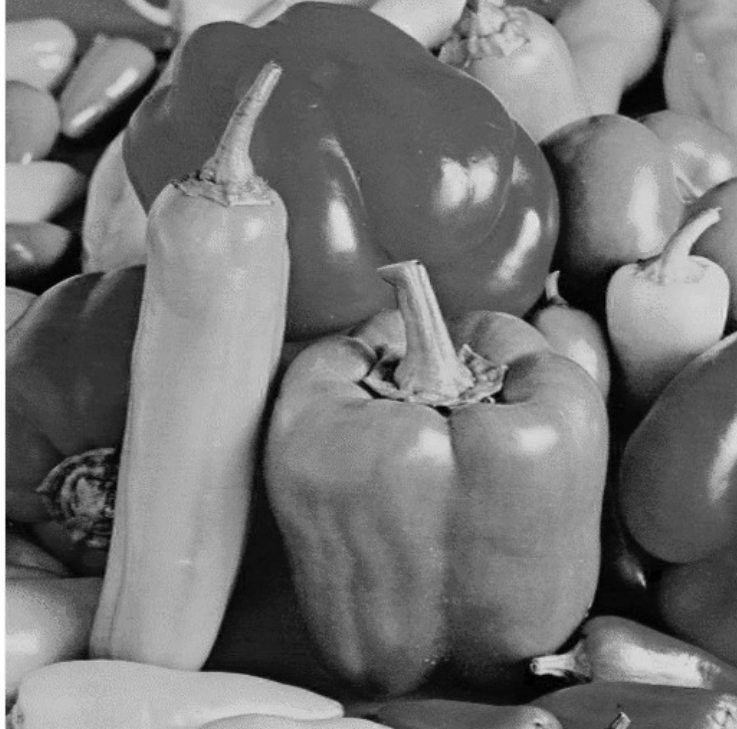
Fig. 5.4 shows the watermarked versions of the four images in Fig. 5.3. It is hard to notice any differences between the watermarked images and the original ones. Table 5.3 shows the PSNR values of the watermarked images compared to those of [63, 68]. The algorithm in [65] is designed to produce a fixed PSNR of 42 dB, and in [66], the average PSNR is presented as 42.51 dB. From Table 5.3, it can be seen that the proposed algorithm results in higher PSNR in all cases, with an average of 44.13 dB with respect to 40.89, 39.78, 42 and 42.51 dB in [68], [63], [65] and [66] respectively. This shows that the proposed algorithm leads to watermarked images with a significantly better visual quality. This is believed to be because the embedding adapts to the directions of gray level variations which is not the case in the compared methods.



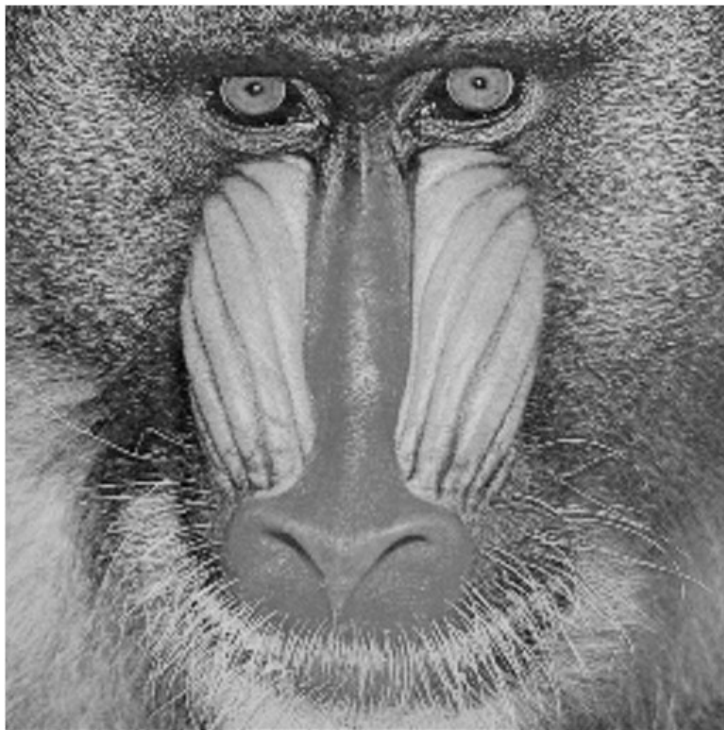
(a) Watermarked Lena



(b) Watermarked Boat



(c) Watermarked Peppers



(d) Watermarked Baboon

Fig. 5.4 Watermarked versions of the images in Fig. 5.3

Table 5.3 Comparison of the PSNR of the watermarked images

Image	PSNR (dB)		
	<i>You [68]</i>	<i>Lee [63]</i>	<i>Proposed</i>
Lena	40.58	41.29	44.34
Boat	39.75	40.54	42.77
Peppers	40.37	41.91	46.29
Baboon	38.41	39.79	43.35
Bridge	N/A	N/A	43.55
Plane	N/A	N/A	44.27
Pirate	N/A	N/A	44.33
<b>Average</b>	39.78	40.89	44.13

### 5.3.2 Robustness of the Watermark

To determine the robustness of the watermark in the proposed algorithm, the watermarked images are modified by different processes then the watermark is extracted and compared to the original watermark.

Fig. 5.6 shows the extracted watermarks from the watermarked versions of the test image Lena after several attacks. The watermark seems to be successfully extracted with good accuracy after JPEG compression even at a low quality of 30%. To better illustrate the robustness of the proposed algorithm to JPEG compression, the average correctness rate, CR, calculated from seven test images is compared to those reported in [61, 63, 65, 66, 68] at different quality factors (QF). The comparison is shown in Table 5.4 and as can be observed, the proposed algorithm performance is above average for the whole range with only the method in [65] yielding higher CR values, however, it must be kept in mind that the PSNR of the watermarked image in the proposed method

is 2.1 dB higher in average which is a considerable improvement. The CR values of the proposed algorithm are very high even at very low quality factors.

JPEG (Quality: 80%)	JPEG (Quality: 50%)	JPEG (Quality: 30%)
Salt and Pepper Noise (Density =0.01)	Salt & Pepper Noise (Density =0.05)	Gaussian Noise (Variance = 0.01)
Brightness Adjustment (+50%)	Brightness Adjustment (-50%)	Gaussian Filter (Standard deviation = 0.5)
Contrast Adjustment (+50%)	Contrast Adjustment (-50%)	Contrast Adjustment (+30%)

Fig. 5.5 The extracted watermarks from the watermarked version of Lena after several attacks

Table 5.4 Comparison of CR at different JPEG quality factors

QF	Correctness Rate (%)						
	<i>You [68]</i>	<i>Lee [63]</i>	<i>Zhi-1 [61]</i>	<i>Zhi-2 [61]</i>	<i>Wan [65]</i>	<i>Zong [66]</i>	<i>Proposed</i>
90	N/A	N/A	99	100	N/A	99.9	99
80	97.35	97.41	95	98	N/A	N/A	98
70	N/A	N/A	72	85	N/A	88.8	97
60	95.48	95.94	71	80	99.8	N/A	96
50	95.01	97.71	70	74	99.3	73.56	95
40	N/A	N/A	30	70	98	N/A	93
30	N/A	N/A	45	38	94	N/A	89
20	N/A	N/A	N/A	N/A	80	N/A	82
10	N/A	N/A	N/A	N/A	N/A	N/A	76

The watermark also survives Gaussian filtering, brightness adjustments. It seems more robust to contrast decrement than increment, nevertheless, the extracted watermark is recognizable in both cases. However, the watermark seems fairly susceptible to noise as it is not retrieved correctly after Gaussian and Salt and Pepper noises. Regardless of susceptibility to noise, the watermark seems robust to the other attacks, most significantly to JPEG compression. This means that this scheme allows the storage and sharing of the image which makes it suitable for authentication purposes.

Overall, the proposed algorithm results in a remarkably better visual quality in the watermarked images, which proves the effectiveness of the feature extraction methodology used. The proposed algorithm yields good robustness against many attacks with noise as the only exception. Most significantly, the proposed algorithm is highly robust to JPEG compression which

makes it very suitable for image authentication purposes. Moreover, the algorithm's computational simplicity allows efficient real-time implementations.

## **5.4 Conclusion**

In this chapter, the feature extraction methodology presented in Chapter 3 is applied towards the development of a DCT based image-adaptive watermarking algorithm that allows for the blind extraction of the watermark. Simulation results has shown that the proposed algorithm is superior in terms of the visual quality of the watermarked images and it results in excellent robustness against compression even at low quality factors. In addition to the high performance in these two usually conflicting aspects, the computational simplicity of the proposed algorithm makes it a very good option for image authentication purposes and real-time applications.

# Chapter 6 Conclusions and Future Work

## 6.1 Concluding Remarks

The work presented in this thesis has aimed at the development of a simple image feature extraction methodology and its application in image protection schemes. A DCT based feature extraction methodology has been developed and based on it, two algorithms, one for image self-recovery and the other for image watermarking, have been proposed.

Regarding image feature extraction, the main concerns were computational simplicity and the ability to distinguish different gray level patterns. An analysis has been conducted to identify how image features are represented in the DCT domain. The analysis revealed that the DCT coefficients with the highest amplitudes, referred to as the DCT peaks, are the most critical frequency coefficients in the representation of image features. They are located in the low and medium frequencies, and therefore it is necessary to extend the range used in feature extraction to include medium frequencies rather than the lowest frequencies only.

In the proposed feature extraction methodology, image blocks containing significant gray level variations, i.e. texture blocks, are identified by their higher ratio of the highest DCT amplitude to the DC component. Then texture classification is applied exclusively to the identified texture blocks. Since blocks of  $8 \times 8$  pixels are small enough to contain only simple gray level patterns, a small set of profiles, thirteen in this work, have been defined and can be used to represent the possible gray level patterns. The texture blocks are classified according to the locations of the highest 3 DCT peaks, so a wider range of frequencies is included without increasing the computational complexity. As a result, the developed methodology is



computationally simple and capable of identifying simple gray level patterns, whereas the existing DCT based feature extraction methods target edge orientations only.

One of the challenges in image self-recovery is to minimize the length of the reference code, in order to facilitate the code embedding it in the cover image without losing the critical image information required to achieve good reconstruction quality. The developed feature extraction methodology has been applied in the proposed image self-recovery algorithm to make the encoding process adaptive to local image features. According to the profile of each block, the critical DCT coefficients representing the specific gray level pattern of the block are chosen and encoded with appropriate numbers of bits. This means the code length assigned to each block is variable depending on its profile. More bits are assigned to the DCT coefficients that are more important to determine the gray level pattern of the block. This way, the encoding process has been made adaptive to the gray level patterns of the image blocks rather to the levels of texture only as in the existing adaptive methods, and consequently, the total number of bits is smaller. Simulation results have shown that the lengths of the reference codes produced by the proposed algorithm are about  $\frac{1}{5} \sim \frac{3}{5}$  of those in the existing methods. This short code requires the use of only 1 LSB plane for embedding, unlike the 2 or 3 LSB planes used by other algorithms. Consequently, the proposed algorithm has resulted in superior visual quality of the embedded images with an average PSNR of 51.64 dB, which is about 7.4~13.7 dB higher than the relevant existing methods. It has also been shown that using the shorter reference codes, as they contain the pertinent image feature information, one can obtain reconstructed images with above average quality at various tampering rates. Overall, the proposed algorithm is computationally simple and very efficient.

Regarding image watermarking, the main challenge is the conflict between the watermark's robustness and the visual quality of the watermarked image. The objective of this

work was to develop a computationally simple algorithm that maximizes the robustness of the watermark without causing visual distortions in the watermarked image and that allows for the blind extraction of the watermark. In order to achieve that objective, by exploiting the properties of the human visual system (HVS), the proposed watermarking algorithm has been made adaptive to image features in three aspects. The first one is that the watermark is embedded in image blocks that contain significant gray level variations so that the resulting alterations are less noticeable. The second aspect is that the DCT coefficients used for embedding are chosen so that the resulting alterations have the same orientation as the dominant gray level pattern. The third aspect is that the embedding strength is adjusted according to the brightness of the block. The developed feature extraction methodology has been applied to detect the gray level variations and their directions and choose the most suitable DCT coefficients for embedding. The simulation results have shown that the proposed watermarking algorithm results in a significantly higher visual quality of the watermarked images than the relevant reported methods with a PSNR that is 2.7 dB higher on average. The proposed algorithm also yields high robustness against JPEG compression even at very low quality factors. This in addition to its computational simplicity, makes it very suitable for image authentication purposes and real-time applications.

In both of the proposed image self-recovery and image watermarking algorithms, adapting to local image features is critical to ensure the quality of performance. The good performances of the proposed algorithms prove the effectiveness of the new DCT-based image feature extraction methodology. This methodology is not confined to these two applications and can be used in a wide range of applications involving the DCT. Furthermore, its computational simplicity makes it useful in hardware and real-time implementations.

## 6.2 Suggestions for Future Work

In the proposed image feature extraction methodology, only the locations of the highest three DCT peaks were considered and not their amplitudes. Inclusion of the amplitudes can help determine if the edges are step or ramp edges, i.e. if the edges are sharp or smooth. Also more profiles can be added to the thirteen profiles and more peaks can be examined for higher accuracy.

In image self-recovery, the reference code used required only a portion of the LSB of pixels in an image to be embedded. This means that the reference code or part of it can be duplicated to increase the chances of recovery in case of severe damage. In addition, more sophisticated encoding techniques can significantly improve the quality of reconstruction and allow the inclusion of more image data in the reference code without a high increase in the length. The inclusion of more texture profiles could also be considered for more precision. Different quality levels can also be included for different levels of texture, for example the texture blocks can be divided into high, medium and low texture blocks and targeting higher reconstruction for higher texture levels.

In Image watermarking, some analysis of the embedding scheme must be conducted to improve robustness to noise. Also, the use of JND models can be investigated in adjusting the embedding strength. The choice of the DCT coefficient for embedding in the proposed method is only based on the dominant orientation of gray level directions. More watermark invisibility might be achieved if the choice depended on the gray level patterns. To do so, the profiles defined in the feature extraction method can be investigated to determine the most suitable choice in each case.

## References

- [1] T. Law, H. Itoh and H. Seki, "Image filtering, edge detection, and edge tracing using fuzzy reasoning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 481-491, 1996.
- [2] T. Ojala, M. Pietikainen and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 971-987, 2002.
- [3] 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 Processing*, vol. 6, pp. 822-830, 2012.
- [4] Z. Su, X. Luo, Z. Deng, Y. Liang and Z. Ji, "Edge-Preserving Texture Suppression Filter Based on Joint Filtering Schemes," *IEEE Transactions on Multimedia*, vol. 15, pp. 535-548, 2013.
- [5] Hyun Sung Chang and Kyeongok Kang, "A compressed domain scheme for classifying block edge patterns," *IEEE Transactions on Image Processing*, vol. 14, pp. 145-151, 2005.
- [6] J. Vega-Pineda, J. Rivera-Mejia, R. Sandoval-Rodriguez and G. Trujillo-Schiaffino, "Acceleration with FPGA for blocks and subblocks edge pattern classification in DCT domain images," in *Proc. IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, May 2014, pp. 707-712.
- [7] X. Wang, H. Wang and Y. Huang, "Fast Block Edge Direction Analysis in DCT domain," in *Proc. International Conference on Wireless Communications and Signal Processing (WCSP)*, Oct. 2010, pp. 1-5.

- [8] B. Shen and I.K. Sethi, "Direct feature extraction from compressed images," in *Storage and Retrieval for Still Image and Video Databases IV*, pp. 404-14, 1996.
- [9] Hongliang Li, Guizhong Liu and Yongli Li, "An effective approach to edge classification from DCT domain," in *Proc. International Conference on Image Processing*, Sept. 2002, vol.1 pp. 940-943.
- [10] M. Eom and Y. Choe, "Fast extraction of edge histogram in dct domain based on mpeg7," in *Proc. of world academy of science, engineering and technology*, 2005, pp. 209.
- [11] J. Jiang, K. Qiu and G. Xiao, "A Block-Edge-Pattern-Based Content Descriptor in DCT Domain," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, pp. 994-998, 2008.
- [12] D.K. Park, Y.S. Jeon and C.S. Won, "Efficient use of local edge histogram descriptor," in *Proc. of the ACM workshops on Multimedia*, Oct. 2000, pp. 51-54.
- [13] I.J. Cox, J. Kilian, F.T. Leighton and T. Shamoan, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions On Image Processing*, vol. 6, pp. 1673-1687, 1997.
- [14] A.J. Ahumada and H.A. Peterson, "A visual detection model for DCT coefficient quantization," in *Computing in Aerospace*, pp. 314-318, 1993.
- [15] H. Peterson, "DCT basis function visibility in RGB space," *Society for Information Display Digest of Technical Papers*, 1992.
- [16] H.A. Peterson, H. Peng, J. Morgan and W.B. Pennebaker, "Quantization of color image components in the DCT domain," in *Electronic Imaging'91, San Jose, CA*, May 1991, pp. 210-222.
- [17] A.B. Watson, J.A. Solomon, A.J. Ahumada Jr and A. Gale, "Discrete cosine transform (DCT) basis function visibility: effects of viewing distance and contrast masking," in

- IS&T/SPIE International Symposium on Electronic Imaging: Science and Technology*, Feb. 1994, pp. 99-108.
- [18] A.B. Watson, J.A. Solomon and A. Ahumada, "Visibility of DCT basis functions: Effects of display resolution," in *Proc. Data Compression Conference (DCC'94)*, Mar. 1994, pp. 371-379.
- [19] A.B. Watson, J. Solomon, A. Ahumada and A. Gale, "Visibility of DCT quantization noise: Effects of display resolution," in *SID International Symposium Digest of Technical Papers*, 1994, pp. 697-697.
- [20] N. Jayant, J. Johnston and R. Safranek, "Signal compression based on models of human perception," *Proceedings of the IEEE*, 1993, vol. 81, pp. 1385-1422.
- [21] A.B. Watson, "DCT quantization matrices visually optimized for individual images," in *IS&T/SPIE's Symposium on Electronic Imaging: Science and Technology*, Sep. 1993, pp. 202-216.
- [22] Z. Wei and K. N. Ngan, "Spatio-Temporal Just Noticeable Distortion Profile for Grey Scale Image/Video in DCT Domain," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, pp. 337-346, 2009.
- [23] J. Wu, G. Shi, W. Lin, A. Liu and F. Qi, "Just Noticeable Difference Estimation for Images With Free-Energy Principle," *IEEE Transactions on Multimedia*, vol. 15, pp. 1705-1710, 2013.
- [24] S. H. Bae and M. Kim, "A Novel DCT-Based JND Model for Luminance Adaptation Effect in DCT Frequency," *IEEE Signal Processing Letters*, vol. 20, pp. 893-896, 2013.

- [25] S. Bae and M. Kim, "A new DCT-based JND model of monochrome images for contrast masking effects with texture complexity and frequency," in *Proc. IEEE International Conference of Image Processing*, Sep. 2013, pp. 431-434.
- [26] S. H. Bae and M. Kim, "A Novel Generalized DCT-Based JND Profile Based on an Elaborate CM-JND Model for Variable Block-Sized Transforms in Monochrome Images," *IEEE Transactions on Image Processing*, vol. 23, pp. 3227-3240, 2014.
- [27] E. Esen and A. A. Alatan, "Robust Video Data Hiding Using Forbidden Zone Data Hiding and Selective Embedding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, pp. 1130-1138, 2011.
- [28] G. L. Wu, T. H. Wu and S. Y. Chien, "Algorithm and Architecture Design of Perception Engine for Video Coding Applications," *IEEE Transactions on Multimedia*, vol. 13, pp. 1181-1194, 2011.
- [29] H. R. Wu, A. R. Reibman, W. Lin, F. Pereira and S. S. Hemami, "Perceptual Visual Signal Compression and Transmission," *Proceedings of the IEEE*, vol. 101, pp. 2025-2043, 2013.
- [30] Z. Luo, L. Song, S. Zheng and N. Ling, "H.264/Advanced Video Control Perceptual Optimization Coding Based on JND-Directed Coefficient Suppression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, pp. 935-948, 2013.
- [31] W. Wan, J. Liu, J. Sun, X. Yang, X. Nie and F. Wang, "Logarithmic Spread-Transform Dither Modulation Watermarking Based on Perceptual Model," in *Proc. IEEE International Conference on Image Processing*, Sep. 2013, pp. 4522-4526.
- [32] S. J. Park, G. Jeon and J. Jeong, "Deinterlacing algorithm using edge direction from analysis of the DCT coefficient distribution," *IEEE Transactions on Consumer Electronics*, vol. 55, pp. 1674-1681, 2009.

- [33] J. Fridrich and M. Goljan, "Images with self-correcting capabilities," in *Proc. International Conference on Image Processing, (ICIP 99)*, Oct. 1999, vol. 3, pp. 792-796.
- [34] X. Zhang, S. Wang and G. Feng, "Fragile watermarking scheme with extensive content restoration capability," in *International Workshop on Digital Watermarking*, Aug. 2009, pp. 268-278.
- [35] X. Zhang, Z. Qian, Y. Ren and G. Feng, "Watermarking With Flexible Self-Recovery Quality Based on Compressive Sensing and Compositive Reconstruction," *IEEE Transactions on Information Forensics and Security*, vol. 6, pp. 1223-1232, 2011.
- [36] P. Korus and A. Dziech, "Efficient Method for Content Reconstruction With Self-Embedding," *IEEE Transactions on Image Processing*, vol. 22, pp. 1134-1147, 2013.
- [37] S. Sarreshtedari and M. A. Akhaee, "A Source-Channel Coding Approach to Digital Image Protection and Self-Recovery," *IEEE Transactions on Image Processing*, vol. 24, pp. 2266-2277, 2015.
- [38] P. Korus and A. Dziech, "Adaptive Self-Embedding Scheme With Controlled Reconstruction Performance," *IEEE Transactions on Information Forensics and Security*, vol. 9, pp. 169-181, 2014.
- [39] Z. Qian and G. Feng, "Inpainting Assisted Self Recovery With Decreased Embedding Data," *IEEE Signal Processing Letters*, vol. 17, pp. 929-932, 2010.
- [40] L. Zhao, Z. Qian, C. Qin and Y. Xie, "An image self-recovery approach with variable payload," in *Proc. IEEE Conference on Industrial Electronics and Applications*, Jun. 2014, pp. 1254-1257.



- [41] M. El'arbi and C. Ben Amar, "Image authentication algorithm with recovery capabilities based on neural networks in the DCT domain," *IET Image Processing*, vol. 8, pp. 619-626, 2014.
- [42] X. Zhang, S. Wang, Z. Qian and G. Feng, "Reference Sharing Mechanism for Watermark Self-Embedding," *IEEE Transactions on Image Processing*, vol. 20, pp. 485-495, 2011.
- [43] R. Chamlawi, A. Khan and I. Usman, "Authentication and recovery of images using multiple watermarks," *Computer and Electrical Eng.*, vol. 36, pp. 578-584, 2010.
- [44] Y. Huo, H. He and F. Chen, "Alterable-capacity fragile watermarking scheme with restoration capability," *Optical Communications*, vol. 285, pp. 1759-1766, 2012.
- [45] P. Korus and A. Dziech, "Reconfigurable self-embedding with high quality restoration under extensive tampering," in *2012 19th IEEE International Conference on Image Processing*, Sep. 2012, pp. 2193-2196.
- [46] Z. Qian, G. Feng, X. Zhang and S. Wang, "Image self-embedding with high-quality restoration capability," *Digital Signal Processing*, vol. 21, pp. 278-286, 2011.
- [47] C. Qin, C. Chang and P. Chen, "Self-embedding fragile watermarking with restoration capability based on adaptive bit allocation mechanism," *Signal Processing*, vol. 92, pp. 1137-1150, 2012.
- [48] C. Qin, C. Chang and K. Chen, "Adaptive self-recovery for tampered images based on VQ indexing and inpainting," *Signal Processing*, vol. 93, pp. 933-946, 2013.
- [49] F. Bornemann and T. März, "Fast image inpainting based on coherence transport," *Journal of Mathematical Imaging and Vision*, vol. 28, pp. 259-278, 2007.
- [50] N. Nikolaidis and I. Pitas, "Robust image watermarking in the spatial domain," *Signal Processing*, vol. 66, pp. 385-403, 5/28. 1998.

- [51] D. M. Thodi and J. J. Rodriguez, "Expansion Embedding Techniques for Reversible Watermarking," *IEEE Transactions on Image Processing*, vol. 16, pp. 721-730, 2007.
- [52] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 890-896, 2003.
- [53] B. Ou, X. Li, Y. Zhao, R. Ni and Y. Q. Shi, "Pairwise Prediction-Error Expansion for Efficient Reversible Data Hiding," *IEEE Transactions on Image Processing*, vol. 22, pp. 5010-5021, 2013.
- [54] I. C. Dragoi and D. Coltuc, "Local-Prediction-Based Difference Expansion Reversible Watermarking," *IEEE Transactions on Image Processing*, vol. 23, pp. 1779-1790, 2014.
- [55] G. Coatrieux, W. Pan, N. Cuppens-Bouahia, F. Cuppens and C. Roux, "Reversible watermarking based on invariant image classification and dynamic histogram shifting," *IEEE Transactions on Information Forensics and Security*, vol. 8, pp. 111-120, 2013.
- [56] T. Zong, Y. Xiang, I. Natgunanathan, S. Guo, W. Zhou and G. Beliakov, "Robust Histogram Shape-Based Method for Image Watermarking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, pp. 717-729, 2015.
- [57] C.I. Podilchuk and Wenjun Zeng, "Image-adaptive watermarking using visual models," *IEEE Journal On Selected Areas in Communications*, vol. 16, pp. 525-539, 1998.
- [58] Jiwu Huang, Y.Q. Shi and Yi Shi, "Embedding image watermarks in dc components," *IEEE Transactions On Circuits and Systems for Video Technology*, vol. 10, pp. 974-979, 2000.
- [59] M. Cedillo-Hernandez, M. Nakano-Miyatake and H. Perez-Meana, "Robust watermarking to geometric distortion based on image normalization and texture classification," in *Proc. Midwest Symposium on Circuits and Systems (MWSCAS)*, Aug. 2008, pp. 245-248.

- [60] Xiuli Wang, Weihua Xie and Xuan Wang, "Adaptive Watermarking Algorithm of Image Based on the Human Visual System," in *Proc. International Conference on Business Computing and Global Informatization (BCGIN)*, Dec. 2012, pp. 511-514.
- [61] Shuang Zhi, Yana Zhang, Cheng Yang and Jianbo Liu, "Edge detection based JND model for digital watermarking," in *Proc. International Conference on Signal Processing (ICSP)*, Oct. 2014, pp. 875-879.
- [62] OuJun Lou, Li Shaohua, Liu ZhaoXia and Tang ShuangTong, "A Novel Multi-Bit Watermarking Algorithm Based on HVS," in *Proc. International Symposium on Parallel Architectures, Algorithms and Programming (PAAP)*, Mar. 2014, pp. 278-281.
- [63] Jung-San Lee and Bo Li, "Self-Recognized Image Protection Technique that Resists Large-Scale Cropping," *IEEE MultiMedia*, vol. 21, pp. 60-73, 2014.
- [64] J. Varghese, O. B. Hussain, B. Babu, J. M. Basheer, S. Subash, M. R. Saadi and M. S. Khan, "An efficient DCT-SVD based algorithm for digital image watermarking," in *Proc. International Carnahan Conference on Security Technology (ICCST)*, Oct. 2014, pp. 1-6.
- [65] W. Wan, J. Liu, J. Sun, C. Ge and X. Nie, "Logarithmic STDM watermarking using visual saliency-based JND model," *Electronics Letters*, vol. 51, pp. 758-760, 2015.
- [66] T. Zong, Y. Xiang, S. Guo and Y. Rong, "Rank-Based Image Watermarking Method With High Embedding Capacity and Robustness," *IEEE Access*, vol. 4, pp. 1689-1699, 2016.
- [67] I. Nasir, F. Khelifi, J. Jiang and S. Ipson, "Robust image watermarking via geometrically invariant feature points and image normalisation," *IET Image Processing*, vol. 6, pp. 354-363, 2012.

- [68] Xinge You, Liang Du, Yiu-ming Cheung and Qiuhui Chen, "A Blind Watermarking Scheme Using New Nontensor Product Wavelet Filter Banks," *IEEE Transactions On Image Processing*, vol. 19, pp. 3271-3284, 2010.
- [69] K. Ramanjaneyulu and K. Rajarajeswari, "Wavelet-based oblivious image watermarking scheme using genetic algorithm," *IET Image Processing*, vol. 6, pp. 364-373, 2012.
- [70] K. Zebbiche and F. Khelifi, "Efficient wavelet-based perceptual watermark masking for robust fingerprint image watermarking," *IET Image Processing*, vol. 8, pp. 23-32, 2014.
- [71] N. B. Halima, M. A. Khan and R. Kumar, "A novel approach of digital image watermarking using HDWT-DCT," in *Proc. Global Summit on Computer & Information Technology (GSCIT)*, Jun. 2015, pp. 1-6.
- [72] B. Mathon, F. Cayre, P. Bas and B. Macq, "Optimal Transport for Secure Spread-Spectrum Watermarking of Still Images," *IEEE Transactions on Image Processing*, vol. 23, pp. 1694-1705, 2014.
- [73] C. C. Lai and C. C. Tsai, "Digital Image Watermarking Using Discrete Wavelet Transform and Singular Value Decomposition," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, pp. 3060-3063, 2010.
- [74] J. Varghese, O. B. Hussain, B. Babu, J. M. Basheer, S. Subash, M. R. Saadi and M. S. Khan, "An efficient DCT-SVD based algorithm for digital image watermarking," in *Proc. International Carnahan Conference on Security Technology (ICCST)*, Oct. 2014, pp. 1-6.
- [75] M. Andalibi and D. M. Chandler, "Digital Image Watermarking via Adaptive Logo Texturization," *IEEE Transactions on Image Processing*, vol. 24, pp. 5060-5073, 2015.
- [76] I.J. Cox, *Digital watermarking and steganography*, Burlington, MA: Elsevier/Morgan Kaufmann Publishers, 2008.

- [77] M. Hamid and C. Wang, "A simple image-adaptive watermarking algorithm with blind extraction," in *Proc. International Conference on Systems, Signals and Image Processing (IWSSIP)*, May 2016.