وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
جامعة سطيف -1
Université SETIF -1

THÈSE

Présentée à la :

FACULTE DE TECHNOLOGIE
DEPARTEMENT D'ELECTRONIQUE

Pour l'Obtention du Diplôme de

Doctorat en Sciences

Par

Ali Ben Ziane

Thème

BLIND IMAGE WATERMARKING USING DISCRETE COSINE AND DISCRETE WAVELET TRANSFORMS

Soutenue le : …….....

Devant le Jury composé de :

Président :	M. Abdelhak FERHAT-HAMIDA	Prof.	Université de Sétif -1
Rapporteur :	M. Noureddine BOUCENNA	M.C.A.	Université de Sétif -1
Co-Rapporteur :	M. Khier BENMAHAMMED	Prof.	Université de Sétif -1
Examinateurs :	M. Abdelhani BOUKROUCHE	Prof.	Université de Guelma
	M. Farid BOUTTOUT	Prof.	Université de B. Bou-Arréridj
	M. Slami SAADI	M.C.A.	Université de Djelfa.

Acknowledgement

I would like to express my earnest gratitude to my supervisor, Dr. Noureddine Boucenna, for his invaluable guidance and stimulation throughout this research work. I greatly appreciate his patience and confidence in my research ability.

I would also like to express my appreciation to my co-supervisor Prof. Khier Benmahammed for his help, advice and discussions.

I wish to thank Prof. Abdelhak Ferhat-Hamida for his support, advices and for accepting to be the head of my examination committee.

I also want to express my deep gratitude to Prof. Abdelhani Boukrouche, Prof. Farid Boutout and to Dr. Slami Saadi for taking the time and effort to read and examin my thesis.

Finally, I would like to dedicate this modest thesis to my beloved mother, my father, my wife and to all members of my family.

Contents

List of Tables

List of Figures

Chapter 1

GENERAL INTRODUCTION

The process of embedding a watermark in a host signal is termed as digital watermarking. The host signal can be a video, audio, image, 3D mesh, etc., while the watermark can be a logo, image, serial number, owner's ID, name, or any other information which shows ownership of the host signal [**Potdar2005**; **Mousavi2014**]. These signatures are normally converted into a binary sequence before being embedding into the host signal. Content providers want to embed watermarks in their multimedia objects (digital content) for several reasons like copyright protection, content authentication, tamper detection, etc [**Woo2007**; **Le2010**].

More and more researchers are particularly attracted to the area of image watermarking because of the property of the image as it has a lot of redundant information contained in it which can be exploited to be used for watermark embedding. The embedding process is guided by the use of a secret key which decides the locations within the image where the watermark would be embedded. When the owner wants to check the watermarks in the possibly attacked and distorted digital images, s/he relies on the secret key that was used to embed the watermark. Using the secret key, the embedded watermark sequence can be extracted.

In order to be successful, the watermark should be invisible and robust against common image processing operations such as additive noise, compression, cropping, filtering, resizing, etc [**Song2010**]. The robustness against image distortion is better achieved if the watermark is placed in the perceptually significant coefficients of the image. These coefficients do not change much after common image processing and compression operations. Also, if these coefficients are destroyed, the reconstructed image is different from the original image and the digital watermark become irrelevant. Although, embedding the watermark in perceptually significant coefficients could alter the perceived visual quality of the image. Thus, two essential

prerequisites for a powerful watermarking scheme, robustness and invisibility conflict with each other [**Mousavi2014**].

Watermarking techniques can be broadly categorized into two distinct categories: non-blind or blind depending on whether the original image is necessary for watermark extraction or not. In real-world practices, non-blind watermarking algorithms are unsuitable for many practical applications in that they require the non-watermarked data to be presented during extraction or detection [**Abdallah2011**].

Watermarking techniques can be also classified according to the domain in which the watermark is embedded, i.e., the spatial domain or the transform domain. While the spatial domain techniques are having least complexity and high payload, they can not withstand image compression and other common image processing attacks [**Potdar2005**]. Transform domain watermarking schemes like those based on the discrete Fourier Transform (DFT) [**Pun2006**; **Solachidis2001**], the discrete cosine transform (DCT) [**Hernandez2000**; **Chu2003**] and the discrete wavelet transform (DWT) [**Barni2001**; **Lu2012**], typically provide higher image imperceptibility and are much more robust to image manipulations. However, DWT has been used more frequently in digital image watermarking due to its time/frequency decomposition characteristics, which resemble to the theoretical models of the human visual system [**Barni2001**].

In essence, most of the existing schemes demonstrate robustness against some categories of attacks but fail to perform well against other types of attacks. Therefore, the challenge remains in the field of digital watermarking in designing schemes that are more robust against a broad range of attacks and maintain reasonable image quality. With the aim to provide higher security and robustness, a digital watermarking scheme is proposed in the present thesis that not only outperforms other methods with respect to various attacks for most of the cases, but also maintains a satisfactory image quality.

In this thesis, we will propose a new blind and robust image watermarking scheme based on discrete wavelet transform (DWT) and discrete cosine transform (DCT). Two DCT-transformed sub-vectors are used to embed the bits of the watermark sequence in a differential manner. The original sub-vectors are obtained by the sub-sampling of the approximation coefficients of the DWT transform of the host image. During the extraction stage, the simple difference between the corresponding sub-vectors of the watermarked image, gives directly the embedded watermark sequence.

To further emphasize the efficiency of combining DWT and DCT domains, we propose also a reduced DCT-based version of our method which is based on the DCT domain only. The

results of comparison between the combined DWT-DCT and the DCT-only methods justify the combination of DWT and DCT domains.

After that, we will extend the outcomes of this research to color video watermarking using a frame-by-frame embedding strategy. To do so, we perform a color space conversion in order to select the most appropriate channel for watermark embedding.

At the end of the thesis, we will propose an implementation of the video watermarking method using the Raspberry Pi platform. In order to further secure the system, we will use a biometric feature (Fingerprint) as a watermark instead of an arbitrary binary image.

The remaining parts of this thesis are organized as follows: Chapter 2 provides the basic concepts of digital image watermarking, its main properties and its applications. It gives also an overview on the classification of different watermarking techniques.

Chapter 3 discusses the proprieties of the DWT and the DCT transforms. After that, it provides a short literature review of the existing watermarking methods operating in these transform domains.

In Chapter 4, we will detail the two proposed watermarking methods. Simulation results and comparison with other existing schemes are also given.

In Chapter 5, we will focus on the extension of the proposed techniques to color video watermarking. We will show also the results of simulation with regard to the transparency and the robustness of the watermarked video segments.

Chapter 6 will discusses the implementation of the video watermarking method using the Raspberry Pi platform. Some results about the efficiency of the implementation are also given at the end of the chapter.

Finally, conclusion about the achievements of this work will be drawn in Chapter 7. Suggestions for future research directions will also be discussed.

Chapter 2

Digital Image Watermarking

2.1 Introduction

Advances in the development of digital data and the Internet have resulted in changes in the modern way of communication. A digital multimedia content, as opposed to an analog one, does not lose quality due to multiple copying processes. However, this advantage of digital media is also their major disadvantage in terms of copyright and the unauthorized use of data [**Stankovic2012**]. Cryptographic methods and digital watermarking techniques have been introduced in order to protect the digital multimedia content. Cryptography is used to protect the content during transmission from sender to recipient.

On the other hand, digital watermarking techniques embed secret information (i.e. watermark) into digital multimedia data such as texts, audio, images, and video by taking into account the limitations of the human perception system such as Human Auditory System (HAS) and Human Visual System (HVS) [**Cox2008**]. The watermark is hidden in the host signal in such a way that it is inseparable from the host signal and so that it is resistant to any signal processing operation not degrading the host signal. Thus, after watermarking, the multimedia data (host signal) is still accessible but permanently marked [**Cox2008**].

2.2 Basic Concepts in Watermarking Scheme

2.2.1 Overview of a Data Security System

The watermarking concept is closely related to two other fields: cryptography and steganography (Figure 2.1). These areas fall under the domain called data security system. Cryptography

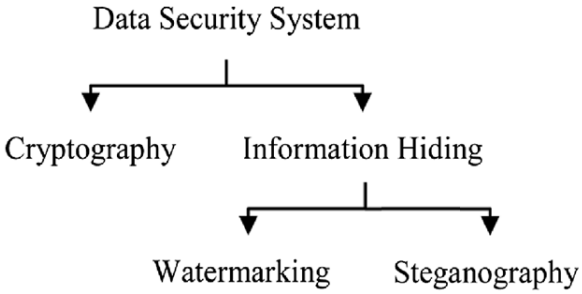


Figure 2.1: Overview of a data security system [**Cheddad2010**]

is a method for sending a message in a secure format that only the authorized person can decode and read [**Cheddad2010**]. This is known as a "secret writing". Even though the encrypted message can be protected during the transmission, once the message is decrypted, it is not protected anymore, and this is the main shortcoming of cryptography techniques when compared with watermarking [**Mousavi2014**].

Steganography is derived from the Greek word "steganos" and "graphei" which mean "covered" and "writing," respectively. In spite of some similarities between steganography and watermarking, there are some differences between them as explained below [**Mousavi2014**; **Subhedar2014**]:

- The objective of steganography is to embed an unrelated secret message into a cover work, while in watermarking, the embedded information and cover work are related to each other.
- In steganography, the message should be invisible, but in watermarking, the embedded information can be either visible or invisible.
- The main goal of steganography is to hide the message into the cover data in a way so that an invader cannot detect it, while the main purpose of watermarking is to embed the data into the cover data in a way that it cannot be removed or replaced by an intruder [**Cheddad2010**; **Subhedar2014**].

Based on these pros and cons, it can be concluded that watermarking is the best choice for preserving the security of a digital image. In addition, the data can be encrypted before embedding the watermark, as a second layer of protection [**Mousavi2014**].

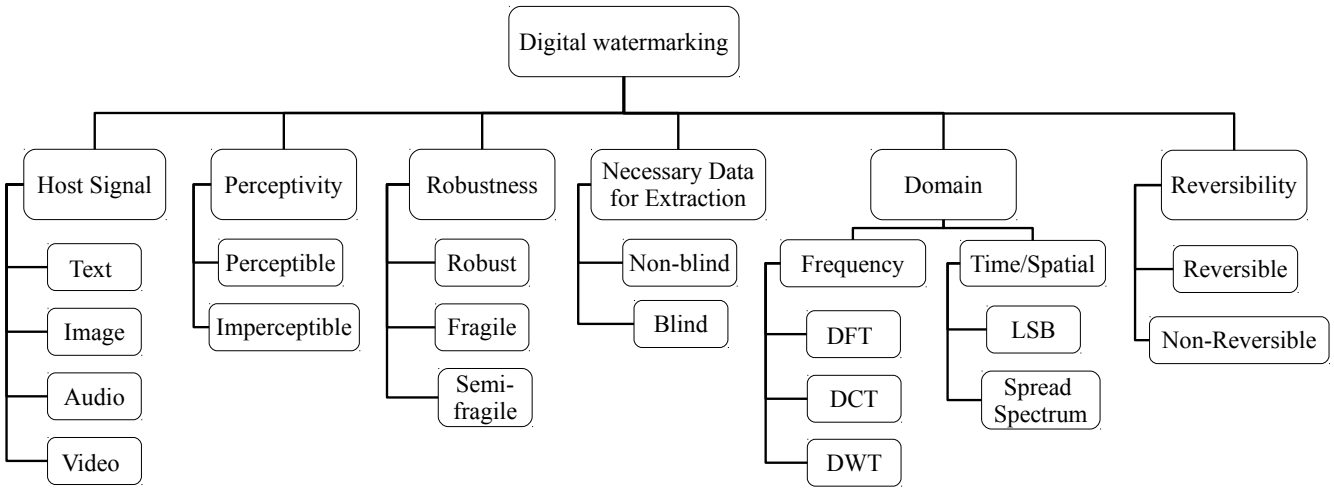


Figure 2.2: Classification of digital watermarking techniques

2.2.2 Classification of Watermarking Techniques

Watermarking methods can be classified based on different views. Most of them can be classified into one of the categories given in Figure 2.2. Regarding the type of host media, digital watermarking techniques can obviously be divided four different categories: text, audio, image and video watermarking.

According to human perception, the watermarking methods can be grouped into perceptible and imperceptible watermarks. A popular illustration of visible methods is logos, which are put at the corners of images or videos for content or copyright protection. Invisible watermarks are useful for application such as authentication, integrity verification, and copyright protection [**Potdar2005**; **Abdullatif2013**].

Watermarking techniques are also divided into robust techniques, semi-fragile and fragile watermarking techniques. The fragile method allows the watermark to easily be destroyed by the smallest of modifications [**Potdar2005**; **Chang2013**]. Applications for this kind of watermarking are limited to authentication and integrity verification. The semi-fragile method protects the hidden data against intentional attacks, but is fragile against malicious attacks [**Altun2006**; **Wenyin2011**]. The robust watermarking method, which is usually used for copyright protection purpose, should be resistant against multiple different attacks [**Makbol2013**; **Tao2014**]. Robust techniques involve embedding a watermark in the original signal, such that

the watermark removal causes serious degradation of the signal quality. Watermark should be designed to be robust to the standard signal processing approaches (compression, filtering, etc.), as well as to intentional attempts to remove the watermark [**Cox1997**; **Stankovic2012**]. Creating robust watermarking methods is still a challenging research problem. These algorithms are robust against some attacks but not against most of them [**Hu2016**].

Digital image watermark techniques can also be classified based on the type of information needed in the extraction process. Using this classification criterion, it can be classified into two categories; non-blind and blind watermarking [**Mousavi2014**; **Song2009**]. A non-blind watermarking system requires the host image and the watermarked image in order to detect and extract the watermark data, but on the other hand, a blind watermarking system requires nothing other than the watermarked image itself to complete the process [**Potdar2005**; **Abdullatif2013**].

Based on the embedding information concept, watermarking algorithms can be classified as either time/spatial or transform/frequency domain [**Cox2008**; **Bhowmik2010**]. In the spatial domain, the watermark information is directly embedded in the pixel value of the host or cover image. These methods are fast and simple and also provide high capacity for embedding watermarks [**Potdar2005**] . Spatial domain methods may have some advantages and may overcome cropping attacks, but their main drawback is their weaknesses against noise or lossy compression attacks. In addition, upon discovering the method, embedded watermarks can easily be modified by a third party [**Mousavi2014**]. In the transform domain, the watermarked image is obtained by embedding the watermark onto the transformed version of the original image [**Cox2008**]. The most popular algorithm in spatial domain is least significant bits (LSB) [**Lin2005**] whereas the most popular frequency domains are the Discrete Fourier Transform (DFT) [**Solachidis2001**; **Pun2006**], Discrete Cosine Transform (DCT) [**Chu2003**; **Suhail2003**; **Xiao2008**]and Discrete Wavelet Transform (DWT) [**Chen2003**; **Tao2004**; **Serdean2007**; **Yusnita2008**; **Bhatnagar2012**].

In addition to above groupings, reversibility (also known as lossless or invertible watermarking) is another important aspect in watermarking [**Brar2013**; **Yang2013**]. Compared to the conventional watermarking schemes, reversible data hiding restores not only the watermark but also the original multimedia perfectly, which is a critical requirement for medical and military applications [**Mousavi2014**]. The main characteristic of reversible methods is the ability to recover the original image without any distortion after extracting the watermark bits, besides providing tamper proofing and authentication. By using a reversible data hiding algorithm to

embed patient information and diagnostics data into the medical image, medical officers can recover perfectly both the hidden information as well as the image itself [**Mousavi2014**].

2.2.3 Different Parts of a Typical Watermarking System

The process carried out in a typical digital image watermarking system can generally be divided into three distinct parts namely Embedding, Transmission and Extraction as illustrated in Figure 2.3. As an additional protection level, in the embedding process the watermark data may be encoded into the host image using an encryption key [**Cox1997**; **Abdullatif2013**].

- Embedding: In this part, the original image and the proposed watermark enter to the system, and according to the embedding algorithm, the watermarked image will be produced.

- Transmission: Ability of others to access the watermarked image. For instance, it can be sold to the customers or it can be published through the Internet.

- Attacks: Modification of the watermarked image intentionally or unintentionally, by a third party. This concept will be explained in the next section.

- Extraction: Process of separating the hidden information from the watermarked image. Extracting algorithms can be divided into three parts: non-blind, semi-blind, and blind. In non-blind or private watermarking, the original image is required during the extraction process [**Potdar2005**]. The original watermark or other side information is necessary to perform the extraction in semi-blind methods. In blind or public watermarking, the extraction process is done without any side information, original image, or original watermark. After that, the quality of watermarked images and accuracy of extracted watermarks will be evaluated by measuring the similarity between the extracted and the original one [**Abdallah2011**; **Abdullatif2013**].

2.2.4 Common Requirements in Watermarking

Depending on the application and the type of data to be watermarked, the watermarking procedure should fulfill a number of requirements [**Potdar2005**; **Tao2014**]. In the sequel, we discuss some general and very common watermarking requirements.

- **Transparency** (Known also as Imperceptibility or Fidelity): The digital watermark should not affect the quality of the original image after it is watermarked. It refers to

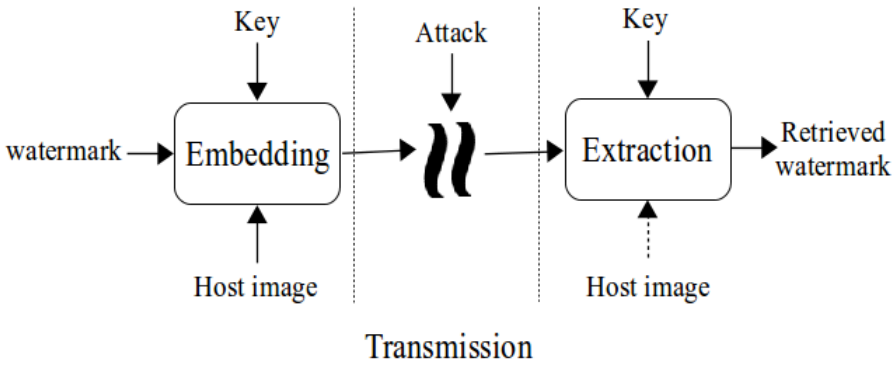


Figure 2.3: A generic image watermarking "information security" scheme. Notice that in blind watermarking schemes, the original image is not needed at the extraction stage [**Cox2008**].

the similarity of original and watermarked images. The watermarks do not create visible artifacts in still images, alter the bit rate of video or introduce audible artifacts in audio signals [**Cox2008**; **Jabade2011**] .

- **Robustness** : The watermarks should not get degraded or destroyed as a result of unintentional or malicious image and signal processing operations [**Potdar2005**]. All operations that may lead to the loss of the watermark information are simply called attacks. Some of the common examples are compression algorithms, filtering, change of the data format, noise, cropping signal samples, re-sampling, etc [**Cox2008**; **Potdar2005**]. Depending on the application, the digital watermarking technique can support different levels of robustness against changes made to the watermarked content .

- **Capacity or Data Payload:** It is the maximum amount of information that can be hidden without degrading image quality. It can be evaluated by the amount of hidden data. This property describes how much data should be embedded as a watermark so that it can be successfully detected during extraction. Watermark should be able to carry enough information to represent the uniqueness of the image. Different application has different payload requirements [**Cox2008**].

The watermark properties explained above have conflicting characteristics as illustrated in Figure 2.4. For example, increasing the robustness of a watermark would normally lower its imperceptibility due to the higher watermark energy imposed on the cover image. In addition, higher

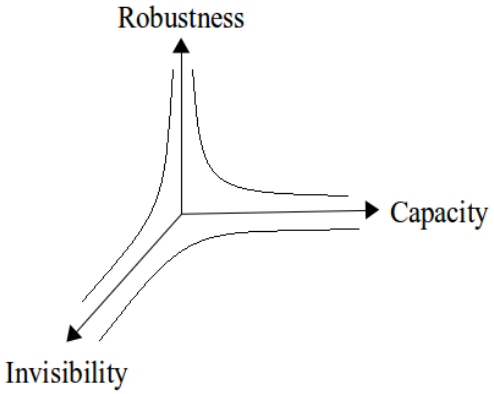


Figure 2.4: Trade-offs between robustness, invisibility and capacity [**Woo2007**].

capacity would compromise its imperceptibility because more modifications the cover image are needed to embed the watermark [**Woo2007**] .

In practice, an image watermarking system would try to balance the right combination of the three measures to achieve its desirable results [**Cox2008**]. Besides the general watermarking requirements discussed above, there could be some specific requirements as well related to the following issues : – Real-time implementation – Complete extraction/reconstruction of the watermark at the decoder – Absence of the original data during the watermark extraction (blind extraction).

2.3 Overview on Watermarking Techniques

As it said before, watermarking algorithms are divided into two main categories: spatial domain and transform domain . The following are brief descriptions of the characteristics of each group.

2.3.1 Spatial Domain Techniques

The most straightforward way to hide a watermark signal within a host signal is to directly embed a watermark in the original host signal. For audio signal, this direct watermarking technique is called time-domain watermarking, whereas for still images this corresponds to spatial-domain watermarking [**Cox2008**]. In the spatial domain, the watermark information is embedded directly in the pixel value of the host or cover image,and to preserve the image quality, the watermark is usually embedded into the least significant bits of the host image [**Lin2005**

]. These methods are fast and simple and provide high capacity for embedding watermarks [**Mousavi2014**]. The other advantage of these techniques is that a small watermark can be embedded several times, so the possibility of removing all watermarks by any kind of attack is very low. Hence, even a single surviving watermarking may fulfill the needs [**Potdar2005**; **Mousavi2014**].

However, spatial domain approaches cannot survive against noise or lossy compression attacks [**Potdar2005**; **Cox2008**]. Furthermore, once the method is uncovered, embedded watermark can be easily modified by a third party. One of the simplest spatial domain techniques is the least significant bit (LSB) method. As shown in Figure 2.5, the input image is firstly binarized by the LSB method. Second, the rightmost bits of each pixel are replaced by input watermark bits. Finally, the modified binary pixel values are converted back to decimal pixel values [**Mousavi2014**].

Another method in the category of spatial domain watermarking is the local binary pattern (LBP) method [**Wenyin2011**; **Chang2013**]. In LBP, watermarking the image is first divided into non-overlapping square blocks. Next, the local pixel contrast is obtained by measuring the spatial relation between the central pixel and its neighboring pixels in each block. These pixels are then used for the embedding and extracting of watermarks according to the rules mentioned in [**Chang2013**]. The advantages of LBP-based methods over LSB methods are their robustness against luminance change, contrast adjustment, and their fragility to other attacks such as filtering and blurring. In other words, LBP-based techniques can be used in semi-fragile watermarking applications [**Wenyin2011**].

Histogram modification [**Tai2009**] is another spatial domain method that takes the global characteristics of the original image into account for embedding watermarks. This scheme tries to shift the values between the minimum and maximum points of the histogram to perform data hiding. A very small amount of side information is generated by this method, and it can also be implemented easily, but the embedding capacity of this method is limited by the number of max points that occur [**Mousavi2014**].

2.3.2 Transform Domain Techniques

In transform domain watermarking, prior to embedding the watermark, transformation such as discrete Fourier transform (DFT) [**Solachidis2001**; **Pun2006**], discrete-cosine transform (DCT) [**Chu2003**; **Xiao2008**; **Ali2014**], discrete-wavelet transform (DWT) [**Chen2003**; **Tao2004**; **Serdean2007**; **Yusnita2008**; **Bhatnagar2012**]., or contourlet [**Rahimi2011**;

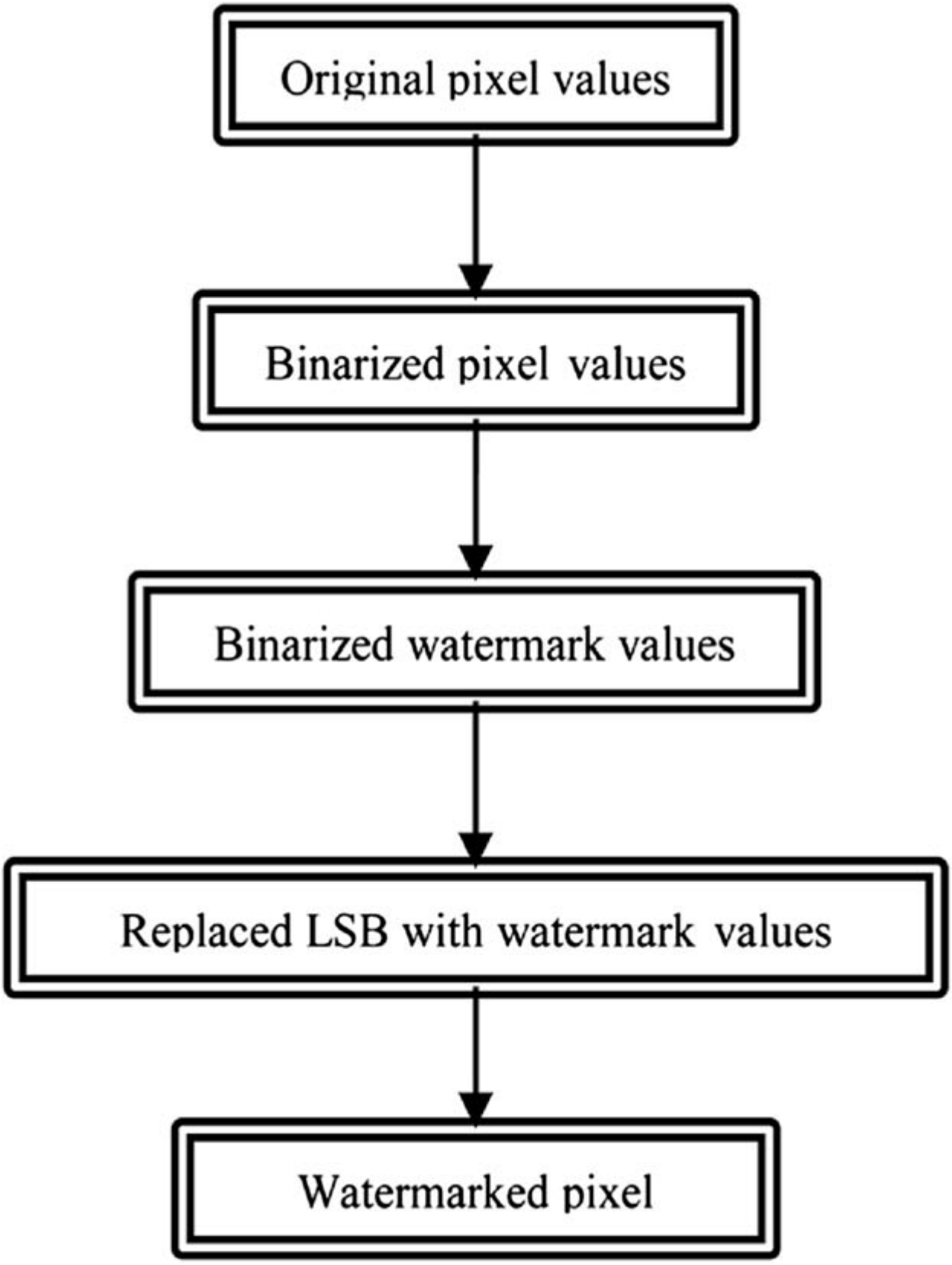


Figure 2.5: LSB watermarking procedure

	Spatial domain	Transform domain
Capacity	High	Low
Robustness	Low	High
Imperceptibility	Highly controllable	Lower controllable
Processing time	Low	High
Complexity	Low	High

Table 2.1: Comparison between spatial and transform domain watermarking methods.

Selvy2013]is applied onto the host image to produce the transformation domain coefficients. The watermarked image is obtained by modifying these transformation coefficients.

Usually, transform-domain watermarking techniques exhibit a higher robustness to attacks. In particular, by spreading the watermark over the whole asset, they are intrinsically more resistant to cropping than spatial domain techniques, where resistance to cropping can only be granted by repeating the watermark across the asset [**BARNI2007**]. Also robustness against other types of geometric transformations, e.g. scaling or shifting, is more easily achieved in a transformed domain, since such a domain can be expressly designed so to be invariant under a particular set of transformations. For instance, techniques operating in the magnitude of DFT domain are intrinsically robust against shifting, since a shift in the time/space domain does not have any impact on DFT magnitude [**BARNI2007**].

Table 2.1 shows some differences between the spatial and transformation domain watermarking techniques with regard to capacity, robustness, imperceptibility, processing time, and complexity [**Mousavi2014**].

2.4 Applications of digital watermarking techniques

Digital watermarking techniques have wide ranging applications. Some of the applications are enlisted below :

- **Copyright Protection**: Digital watermarks can be used to identify and protect copyright ownership. Digital content can be embedded with watermarks depicting metadata identifying the copyright owners [**Cox2008**].
- **Tamper Detection**: Digital content can be detected for tampering by embedding fragile watermarks. If the fragile watermark is destroyed or degraded, it indicated the presence of

tampering and hence the digital content cannot be trusted [**Ouyang2007**]. Tamper detection is very important for some applications that involve highly sensitive data like satellite imagery or medical imagery. Tamper detection is also useful in court of law where digital images could be used as a forensic tool to prove whether the image is tampered or not.

- **Content Archiving:** Watermarking can be used to insert digital object identifier or serial number to help archive digital contents like images, audio or video [**Ouyang2007**]. It can also be used for classifying and organizing digital contents. Normally digital contents are identified by their file names; however, this is a technique as file names can be easily changed. Hence embedding the object identifier within the object itself reduces the possibility of tampering and hence can be effectively used in archiving systems [**Ouyang2007**; **Arya2015**].

- **Meta-data Insertion:** Meta-data refers to the data that describes data. Images can be labeled with its content and can be used in search engines [**Arya2015**]. Meta-data can be also used for the authentication of digitally preserved patient's medical record, including blood sample, X-ray, ECG, EEG etc [**Kumari2013**].

- **Concealed Communication**: Since watermarking is a special technique of steganography, it can be used for concealed communication also [**Wipro2001**].

2.5 Attacks on Watermarking System

In image watermarking techniques, the main consideration is the evaluation of the robustness and effectiveness of the watermarking method through measurement of the impact of different attacks upon the watermarked image [**Mousavi2014**]. The attacks are broadly classified as signal processing attacks and geometric attacks [**Le2010**; **Song2010**]:

- **Signal Processing Attacks**: Signal processing attacks are also called as image processing attacks or non geometric attacks. These common signal processing attacks may include compression of image, addition of noise like Gaussian or salt and pepper noise, gamma correction, filtering, sharpening, histogram equalization, etc.

- **Geometric Attacks:** Geometric attacks include basic geometric transformations in an image. These include geometrical distortions like rotation, scaling, translation, cropping, row-column blanking, warping etc. Geometric attacks attempt to destroy synchronization of detection thus making the detection process difficult and even impossible.

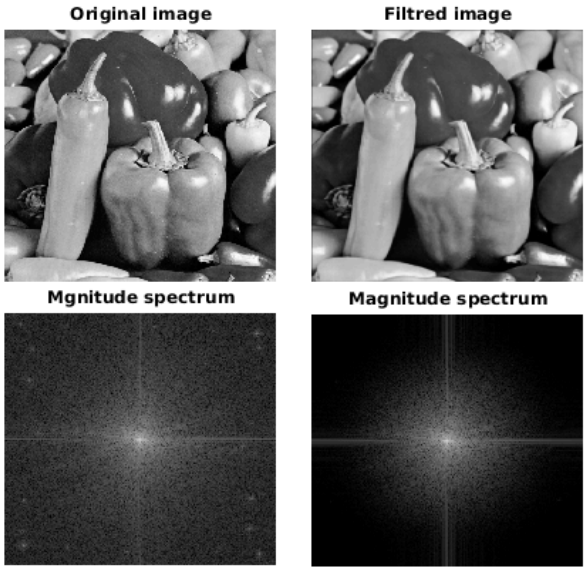


Figure 2.6: The effect of low-pass filtering with an average 3×3 filter on the spectrum of the image.

Based on the watermarking method used, the image may be robust against a specific group of attacks. For instance, in order to increase robustness against geometrical attacks, Fourier-based methods may be a good solution.

2.5.1 Examples of image attacks

- **Filtering attacks:** Low pass filtering like spatial averaging and Gaussian filtering, can affect the performance of the watermark detection procedure since the spectrum of watermarks often have significant high frequency spectral content as show in Figure 2.6. An attacker can use this in an attempt to remove a watermark [**Cox2008**; **Le2010**]. For example, a watermark with significant energy in high frequencies might be degraded by the application of a low-pass filter. Furthermore, any watermarking system for which the added pattern is 'noise-like' is susceptible to noise removal techniques [**Cox2008**; **Ahmed2013**].

- **Additive noise:** It can be produced by the sensor and circuitry of a scanner or digital camera. The attacker may introduce shaped noise with maximum unnoticeable power, for example add some noise to the image in the range +1 or -1 and this will destroy any watermark embedded in the spatial domain (least significant bit) [**Ahmed2013**].

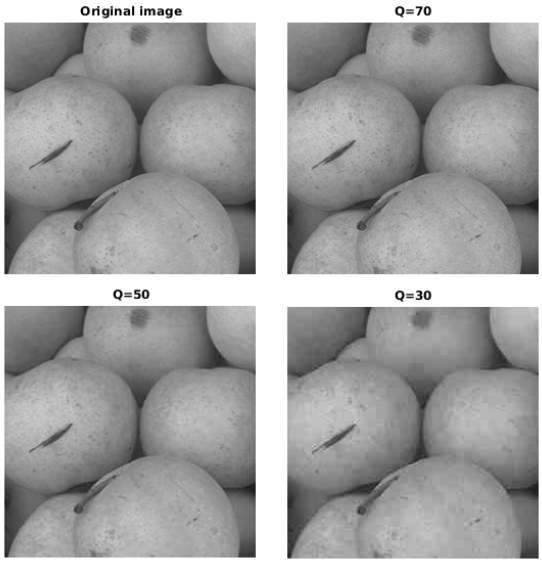


Figure 2.7: JPEG compression for 3 quality values. The compression artifact is not noticeable only at high compression ratio (q=30)

The noise can affect also the detction if the watermark was inserted in the low-frquency contenent of transform domain [**Cox2008**].

- **JPEG compression:** JPEG is a lossy compression format and the key to file size reduction is choosing the maximum amount of compression without affecting the picture quality as shown in Figure 3.7. These losses are due to the quantization process. In general, the ratio of compression and consequently the losses are proportional to the number of quantization levels [**Ahmed2013**]. As the number of quantization levels decreases, the compression ratio increases and the losses increase. This is due to the fact that the quantized coefficient value is rounded to the nearest quantization level. If there is a small number of a quantization level then the coefficient value will be rounded to a value which differs significantly from its original value. Such changes in the coefficients will damage the watermarking content embedded in these coefficients [**Feng2009**].

2.6 Performance evaluation of watermarking algorithms

Performance evaluation is very important part in the any algorithmic design in watermarking. It is measured by comparing the watermarked image ($\hat{I}$) with the original unmarked image (I) and is calculated by various metrics. The most used metrics are mean square error (MSE) and the peak signal to noise ratio (PSNR).

2.6.1 Mean square error (MSE)

The mean squared error (MSE) in an image watermarking is to estimate or measures the average of the squares of the "errors", between host image and watermark image [**Khorrami2014**].

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} \left[I(i,j) - \hat{I}(i,j) \right]^2 \tag{2.1}$$

Where I is an $m \times n$ monochrome input image, and $\hat{I}$ is the watermarked output image.

2.6.2 Peak signal to noise ratio (PSNR)

PSNR (Peak Signal to Noise Ratio) is used to determine the Efficiency of Watermarking with respect to the noise. The noise will degrade the quality of image. The visual quality of watermarked and attacked images is measured using the Peak Signal to Noise Ratio [**Potdar2005**; **Khorrami2014**]. It is given by :

$$PSNR = 10 \log_{10} \left(\frac{MAX_I}{MSE} \right) \tag{2.2}$$

Where MAX_I is the maximum possible pixel value in the host image.

Imperceptibility of image is determined by this factor. More the PSNR shows that Watermarked image is perceptible or watermark is not recognized by naked eyes.

2.6.3 Quality of extracted watermark

To evaluate the quality of extracted binary watermark, the bit error rate (BER) and the bit-correct ratio (BCR) are the most used measures to assess the similarity between the original watermark W and the extracted watermark $\tilde{W}$ [**Feng2009**]. The BER is defined by:

$$BER = \frac{B_{err}}{B_{total}} = \frac{1}{L} \sum_{k=0}^{L-1} W(k) \oplus \tilde{W}(k) \tag{2.3}$$

where B_{err} is the total number of erroneous bits, B_{total} is the total number of bits (L) in the watermark sequence W and $\oplus$ is the XOR operator. If the watermark is extracted without errors the BER value will be 0.

Alternatively, the use of the BCR measure has become common recently [**Huang2011**; **Yang2013**], as it allows for a more detailed scale of values and is defined as the ratio of correct extracted bits to the total number of embedded bits [**Yang2013**] :

$$BCR = \frac{1}{L}\sum_{k=0}^{L-1}\overline{W(k)\oplus\tilde{W}(k)} \times 100\% \tag{2.4}$$

If the watermark is extracted without errors the BCR value will be 100%.

2.7 Conclusion

In this chapter the basic properties and applications of digital watermarking are briefed along with the probable attacks on watermarking including image processing attacks and geometrical ones. Most of the algorithms use the transform domain in embedding, since it has been proven that the transform domain watermarking schemes are much better compared to the spatial domain methods. Another observation is that some methods in the literature are built on blind and some on non-blind techniques. In blind techniques the original image is not required and in non-blind techniques the original image is required to extract the watermarking information.

To propose robust watermarking techniques against such variety of attacks DCT and DWT domains watermarking schemes are selected in this thesis. In the next chapter the state-of-the-art study on these transform domain watermarking schemes are discussed and analyzed.

Chapter 3

DWT and DCT watermarking techniques

3.1 Introduction

As we saw in chapter 2, the frequency domain watermarking schemes are exploited using different frequency domain analysis, namely, the discrete Fourier transform (DFT), the discrete cosine transform (DCT), and the discrete wavelet transform (DWT).

There is a debate among researchers regarding the coefficient selection for embedding withing the DCT and DWT domains. Cox et al [**Cox1997**] advise that the watermark should be embedded in the low frequency coefficients to ensure the robustness. Other researchers suggest that the watermark should be embedded in the mid or high frequency coefficients to reduce the resulted distortions [**BARNI2007**; **Ahmed2013**]. In attempt to reach a trade-off between the different transform techniques in terms of the transparency the watermarked images, and the good resistance to attacks, several hybrid techniques have been proposed [**Tsai2004**; **AI-Haj2007**; **Feng2010**; **Singh2014**; **Hu2016**].

In this chapter we will discuss the background of the DWT transform and the discrete cosine transform DCT. A brief literature review of methods working in these transform domains is also given.

3.2 Wavelet transform

Wavelets can be described as a class of function used to localize a time domain input signal in both space and scaling. A family of wavelets can be developed by defining the mother wavelet, $\Psi(t)$, which is confined in a finite interval. The family members, often referred as daughter wavelets can be defined as [**Strang1997**]:

Figure 3.1: The filter bank approach [**Wang2002**].

$$\Psi_{(a,b)}(t) = \frac{1}{\sqrt{a}}\Psi\left(\frac{t-b}{a}\right) \tag{3.1}$$

where $a > 1$ is the change of scale and $b \in \Re$ is the translation in time. Therefore a continuous input signal $f(t)$ can be represented in wavelet transform as a linear combination of daughter wavelets, $\Psi_{(a,b)}(t)$ and the corresponding wavelet coefficients $f(a,b)$ can be defined as [**Strang1997**]:

$$f(a,b) = \int_{-\alpha}^{\alpha} \Psi_{(a,b)} f(t)dt \tag{3.2}$$

$$= \left\langle \Psi_{(a,b)}, f(t) \right\rangle. \tag{3.3}$$

The above defined Continuous Wavelet Transform (CWT) can be extended to Discrete Wavelet Transform (DWT), which is used in image and video coding applications, including watermarking. In case of DWT, usually the wavelet function ($\Psi_{(a,b)} : a,b \in \mathbb{Z}$) follows dyadic translation (by power of 2) and dilation in Hilbert space and (3.1) is modified to:

$$\Psi_{(a,b)}(t) = 2^{a/2}\Psi(2^a t - b) \tag{3.4}$$

The implementation of DWT is adapted primarily by two different methods: The filter bank approach and the lifting based approach. In this thesis, we only consider the first one because it is the most used in image watermarking techniques [**Jabade2011**].

The filter bank approach consists of two filter banks, one each for the analysis (forward transform) and the synthesis (inverse transform) as shown in Figure 3.1. During the analysis, the input signal is passed through two separate channels, using a low pass filter (h_0) and a high pass filter (h_1) followed by a down sampling operation by a factor of 2, in each channel. The low pass filter data (L) contains the low-frequency content of the signal or the approximation information while the high passed data (H) retains the high-frequency content or the details information of the input signal.

To reconstruct the signal data, the transformed coefficients are first interpolated by an up sampling operation with a factor of 2 and then convolved with synthesis filter banks, g_0 and g_1. The filter coefficients of g_0 and g_1 are obtained from corresponding analysis filters h_0 and h_1,

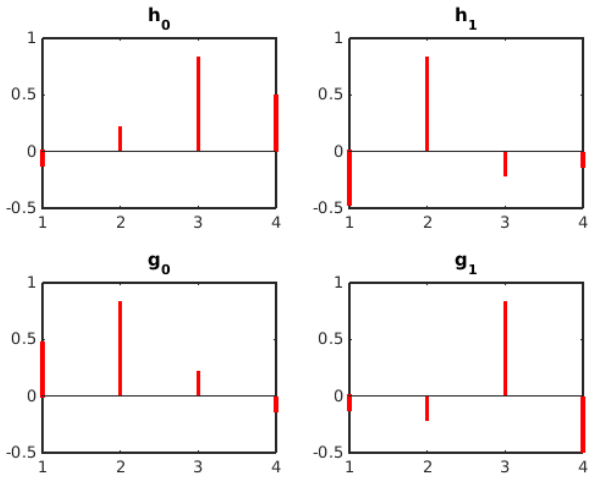


Figure 3.2: The decomposition and reconstruction filters for the wavelet of Daubechies2.

respectively, to eliminate the aliasing. This analysis relies on the concept of quadrature mirror filter (QMF) [**Strang1997**; **Wang2002**].

Let x be a finite energy signal. Two filters f_0 and f_1 are quadrature mirror filters (QMF) if, for any x,

$$\|y_0\|^2 + \|y_1\|^2 = \|x\|^2 \tag{3.5}$$

where y_0 is a decimated version of the signal x filtered with f_0 so y_0 defined by $x_0 = f_0(x)$ and $y_0(n) = x_0(2n)$, and similarly, y_1 is defined by $x_1 = f_1(x)$ and $y_1(n) = x_1(2n)$. This property ensures a perfect reconstruction of the associated two-channel filter banks scheme [**Strang1997**]. Not that the condition in (9) can be formulated as [**Strang1997**] :

$$|F_0(z)|^2 + |F_1(z)|^2 = 1 \tag{3.6}$$

where $F_0(z)$ and $F_1(z)$ are the transfer functions of the filters f_0 and f_1 respectively.

Practically, if we specify the coefficients of the low-pass analysis filter h_0. The other filter coefficients (h_1,g_0,g_1) are typically obtained from h_0. The conjugate quadrature filter specifies h_1 as a reversed version of h_0, with every other value negated [**Weeks2010**]. If $h_0 = [a,b,c,d]$, then $h_1$will be $[d,-c,b,-a]$. For reconstruction, g_0 and g_1 are reversed versions of h_0 and h_1 , respectively. Using the h_0 values from above, we get $g_0 = [d,c,b,a]$, and $g_1 = [-a,b,-c,d]$. For example, the analysis with the wavelet of Daubechies2 [**Strang1997**] uses the four filters shown if Figure 3.2.

Figure 3.3: 2D wavelet transform decomposition [**Wang2002**].

Figure 3.4: 2D wavelet reconstruction [**Wang2002**].

3.3 2D wavelet

The wavelet decomposition of image requires 2D transform and it is achieved by performing 1D DWT separately on rows and columns of 2D signals. At each stage of the transform one low pass (L) and one high pass (H) coefficient subsets are generated and as a result an one level 2D wavelet transform creates four sub-bands, namely, LL, LH, HL and HH as shown in Figure 3.2.a).

The LL sub-band represents the original image in half resolution and contains smooth spatial data with high spatial correlation. The HH sub-band contains the noise and edge information while HL and LH subbands consists of vertically and horizontally oriented high frequency details, respectively. The 2D wavelet transform can repeatedly be applied on LL sub-band from previous decomposition to create hierarchy of the wavelet coefficients. An example of 2 level 2D wavelet decomposition is shown in Figure 3.2.b) [**Bhowmik2010**]. An example of one level DWT decomposition of the image of 'Peppers' using the wavelet of Haar is shown in Figure 3.5.

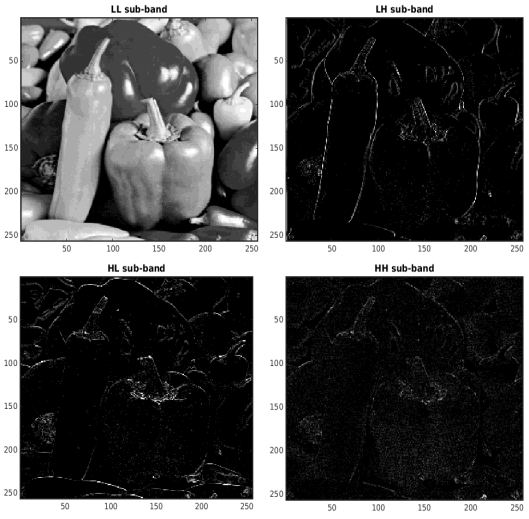


Figure 3.5: 1-level DWT of the image of Peppers.

3.4 The Discrete Cosine Transform

Discrete cosine transform (DCT) is an important transform extensively used in digital image processing. Large DCT coefficients are concentrated in the low frequency region; hence, it is known to have excellent energy compactness properties [**Khayam2003**; **Dutta2014**].

The most common DCT definition of a 1-D sequence x of length N is :

$$c(k) = \alpha(k) \sum_{i=0}^{N-1} x(i) cos \left[\frac{(2i+1)k\pi}{2N} \right] \tag{3.7}$$

for $k = 0, 1, 2, 3...., N-1$. Similarly, the inverse transformation for 1-D sequence of length N is :

$$x(i) = \sum_{k=0}^{N-1} \alpha(k) c(k) cos \left[\frac{(2i+1)k\pi}{2N} \right] \tag{3.8}$$

for $i = 0, 1, 2, 3...., N-1$. In both equations (5) and (6)$\alpha(k)$ is defined as :

$$\alpha(k) = \begin{cases} \sqrt{\frac{1}{N}} & for \quad k = 0 \\ \sqrt{\frac{2}{N}} & for \quad k \neq 0 \end{cases} \tag{3.9}$$

In DCT the first transform coefficient ($c(0) = \sqrt{\frac{1}{N}} \sum_{i=0}^{N-1} x(i)$) is the DC coefficient and all others are the AC coefficients.

The plot of $\sum_{i=0}^{N-1} cos\left[\frac{(2i+1)k\pi}{2N}\right]$ for $N = 8$ and varying values of i is shown in Figure 3.3. In accordance with the previous observation, the first the top-left waveform ($i = 0$) renders a constant (DC) value, whereas, all other wave-forms ($i = 1, 2, ..., 7$) give wave-forms at progressively increasing frequencies [**Khayam2003**]. These wave forms are called the cosine basis function .

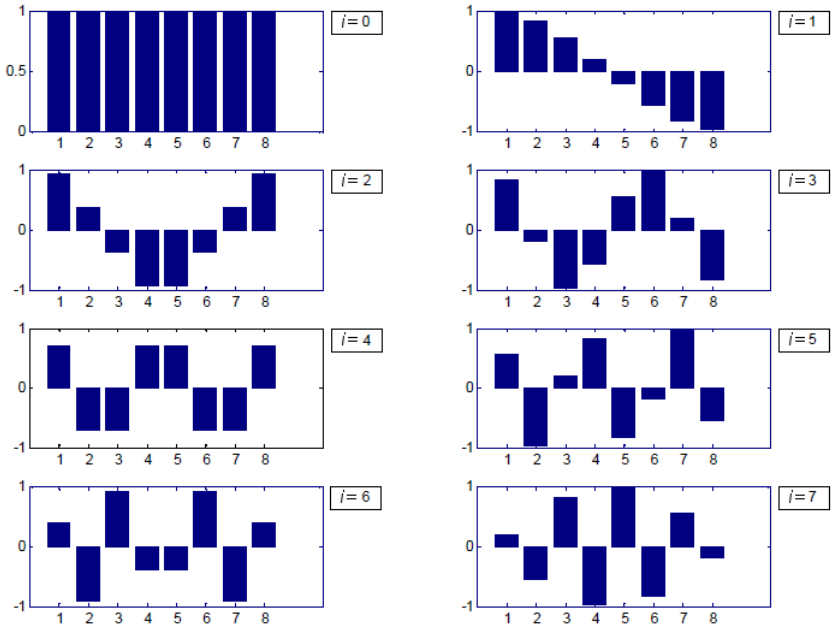


Figure 3.6: One dimensional cosine basis function ($N = 8$) [**Khayam2003**].

Note that these basis functions are orthogonal. Hence, multiplication of any waveform in Figure 3.3 with another waveform followed by a summation over all sample points yields a zero (scalar) value, whereas multiplication of any waveform in Figure 3.3 with itself followed by a summation yields a constant (scalar) value. Orthogonal wave-forms are independent, that is, none of the basis functions can be represented as a combination of other basis functions [**Khayam2003**].

If the input sequence has more than N sample points then it can be divided into sub-sequences of length N and DCT can be applied to these chunks independently. Here, a very important point to note is that in each such computation the values of the basis function points will not change. Only the values of $x(i)$ will change in each sub-sequence. This is a very important property, since it shows that the basis functions can be pre-computed off-line and then multiplied with the sub-sequences. This reduces the number of mathematical operations (i.e., multiplications and additions) thereby rendering computation efficiency [**Khayam2003**].

3.5 Brief survey of DCT and DWT watermarking methods

A lot of DCT-based and DWT-based watermarking methods for still images can be found in the literature. This section will review rapidly some of them. These methods are mainly classified based on their extraction stage requirements as blind detection or non-blind detection.

In 1997, et al. [**Cox1997**] used DCT based spread spectrum communication for multimedia watermarking. This method has become very popular and has been used by many researchers. In this method, a set of independent and identically distributed Gaussian random sequences are embedded in the most perceptually significant frequencies of an image. As in spread spectrum communication the signal energy in any frequency is undetectable if the narrow band signal is transmitted over broader bandwidth. That will cause a watermark to spread over all frequencies so that energy in any single frequency is very small. The Cox method is an incomplete method. Thus, it requires the original image in the extraction process.

Tsai et al. [**Tsai2000**] presented a scalar quantization based blind watermarking scheme which embeds a binary logo as a watermark and blind watermark detection. They embed the watermark in the middle and low frequency components of the wavelet sub-bands i.e. all sub-bands except LL sub band. All the selected coefficients are quantized by a constant factor which is a main issue with this algorithm because certain high texture rich regions within an image can tolerate large modifications (quantization step sizes) because of their inherent high texture masking capacity and hence can be strongly watermarked. At the same time smooth regions have a comparatively lower masking capacity and hence should be quantized using smaller step sizes. This algorithm shows robustness against JPEG compression only.

Tao and Eskicioglu [**Tao2004**] conducted a comparative study to find out the effects of embedding watermarks in the first and second level decomposition. The authors suggested that embedding in the first level is advantageous because it offers more coefficients for modification and the extracted watermarks are more textured and have better subjective visual quality. The technique uses variable scaling parameters for different sub-bands at different decomposition levels. Their main observations are $LL1$ and $LL2$ bands which are robust against JPEG compression, Blurring, Gaussian Noise, Scaling, Cropping, Pixilation and Sharpening. $HH1$ and $HH2$ bands are robust against Histogram Equalization, Intensity Adjustment and Gamma Correction. $HL1, HL2$ and $LH1, LH2$ also show similar robustness. As with the other techniques the main issue with this algorithm is the non-blind nature, where the original image is required for extracting the watermarks

Lin et al. [**Lin2009**] proposed a blind algorithm based on DWT. Cover image is decomposed using fourth-level DWT. The LH4 sub-band is divided into variable size blocks for watermark embedding. In each block of LH4, one bit is embedded by modulating the maximum wavelet coefficient according to the watermark bit. Value of the maximum coefficient is increased for embedding bit 1 and is decreased to embed bit 0. The modified value must be greater than or equal to the second maximum in that block. This requirement is essential for watermark extraction. Some parameters are used to obtain control over the embedding and extraction process. Limitations of this method are low data embedding capacity and the difficulty of adjusting the parameters when there is a change in watermarking requirements or change in input images.

Bhatnagar et al. [**Bhatnagar2012**] proposed non-blind method based on DWT. After the third level decomposition by DWT, watermark is embedded in the selected blocks made by zig–zag sequence. The blocks are selected based on their variances which further serve as the measure of watermark magnitude that could be imperceptibly embedded in each block. Now, the variance calculated in a small moving square window process computes the mean of the standard deviation values derived for the image. The method is highly robust against number of signal processing attacks and time efficient. However, it is less effective for histogram equalization and low-pass filtering attacks.

In [**Das2014**] the authors presented a novel blind watermarking algorithm in DCT domain using the correlation between two DCT coefficients of adjacent blocks in the same position. One DCT coefficient of each block is modified to bring the difference from the adjacent block coefficient in a specified range. The value used to modify the coefficient is obtained by finding difference between DC and median of a few low frequency AC coefficients and the result is normalized by DC coefficient. The authors claim that their algorithm is highly robust against JPEG compression, cropping, noise addition, sharpening, and histogram equalization.

3.6 Combined DWT/DCT watermarking techniques

Further performance improvements in DWT-based digital image watermarking algorithms could be obtained by combining DWT with DCT [**Al-maweri2015**]. The idea of applying two transform is based on the fact that combined transforms could compensate for the drawbacks of each other, resulting in effective watermarking [**AI-Haj2007**]. In this section we explore some combined DWT/DCT watermarking methods for still images.

In the algorithm proposed in [**AI-Haj2007**], the watermarking was carried out through the embedding of the watermark in the first and second level DWT sub-bands of the host image followed by the application of DCT on a selected DWT coefficient sets. It has been shown that the combination of the two transforms improved the watermarking performance considerably when compared to the DWT-Only watermarking approach.

In [**Feng2010**], the authors proposed a blind DWT-DCT watermarking approach. After scrambling the binary watermark, a block-based DCT transform of the first level DWT LL sub-band is computed and two PN-sequences of the watermark bits are embedded in the mid frequency coefficients of the corresponding DCT blocks. In the extraction process, the same steps as the embedding process is used to extract the DCT middle frequencies of the LL sub-band. Finally, correlation between mid-band coefficients and PN-sequences is calculated to determine the watermarked bits.

In [**Deb2012**], the authors proposed a combined DWT-DCT watermarking technique for copyright protection. First, the host image is transformed to the DWT domain using single-level decomposition. The HL sub-band is chosen for the watermark embedding after dividing it into 4×4 blocks and choosing the lowest frequencies coefficients. DCT then is performed for the selected coefficients. The watermark bits are inserted into these coefficients after scrambling the watermark image by Arnold transform. Their method is blind and fulfills the transparency requirement but its robustness is not guaranteed especially against noise adding, low-pass filtering and cropping attacks.

In the paper proposed by [**Hu2016**], the merits of block-based blind watermarking in a composite DWT-DCT domain are explored. To improve the performance in robustness and imperceptibility, the quantization index modulation (QIM) applied to DWT-DCT coefficients has been formulated in an adaptive manner, where controlling parameters are designed to minimize the bit error rates of extracted watermarks subject to a quality criterion. To further enhance watermarking efficiency, two collective strategies are proposed. One takes advantage of multi-bit embedding and the other modifies the norm of a vector constituted by selected coefficients. Experimental results show that their scheme achieves satisfactory improvements in robustness while the imperceptibility is properly maintained.

In [**Hu2016**], the authors developed a block-based blind watermarking technique using in the DWT-DCT domain. After performing the one-level Haar wavelet transform, they selected the middle-frequency range of the LL sub-band to embed the watermark. More specifically, they divided the LL sub-band matrix into non-overlapping blocks of size 4×4 and then convert each block separately using 2-D DCT. By analyzing the statistic properties of DWT-DCT blocks,

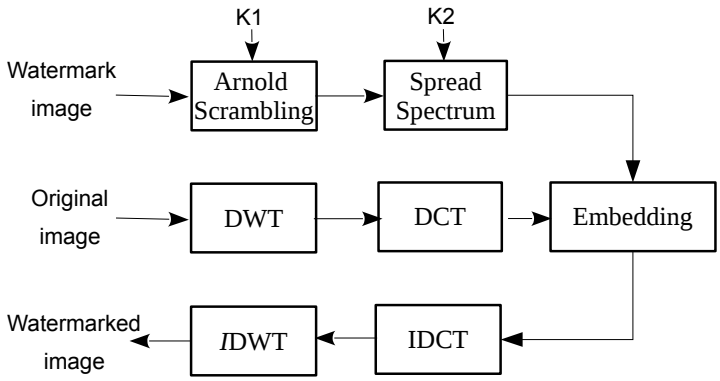


Figure 3.7: Embedding process in Feng et al. method [**Feng2010**].

they selected three AC components for performing watermark embedding based on QIM. The basic idea of QIM is to quantize the coefficient amplitude to an odd index if the watermark bit is "1" and an even index if the watermark bit is "0". After modifying the designated DWT-DCT coefficient, they took the inverse block-based DCT to restore the LL sub-band matrix. The watermarked image was attained by performing one-level inverse DWT with the modified LL sub-band and three original detail sub-bands taken as inputs. Despite its robustness against a number of attacks, this schemes could not withstand geometric distortions.

3.7 Example of combined DWT/DCT methods

In this section we outline one combined blind and robust DWT/DCT method [**Feng2010**], and which we will compare with the proposed method in Chapter 4. After scrambling the binary watermark using Arnold transform, a block-based DCT transform of the first-level DWT LL sub-band is computed and two PN-sequences of the watermark bits are embedded in the mid frequency coefficients of the corresponding DCT blocks. In the extraction process, the same steps as the embedding process is used to extract the DCT middle frequencies of the LL sub-band. Finally, correlation between mid-band coefficients and PN-sequences is calculated to determine the watermarked bits.

The watermarking process is started by applying the DWT to the original image, and subsequently applying the DCT to the DWT sub-bands. The watermark embedding process is represented in Figure 3.5.

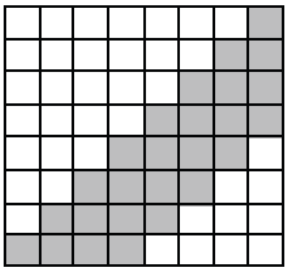


Figure 3.8: DCT coefficient mask for watermark [**Feng2010**].

1. The Original image is decomposed into four sub-bands : *LL*,*HL*, *LH* and *HH* at level 1 in the DWT domain. The sub-band *LL* represents the DWT approximation coefficients while the other sub-bands represent the detail coefficients.

2. Divide the sub-band LL into 8x8 blocks and apply DCT to each block. The watermark is not applied to all block DCT values, but is applied only to the mid frequency DCT coefficients using mask shown in Figure 3.5.

3. Scramble the watermark image with Arnold transform for k1 times and get the scrambled watermark.

4. Generate two uncorrelated pseudo-random sequences: One sequence is used to embed the watermark bit 0 (PN_0) and the other sequence is sued to embed the watermark bit 1 (PN_1). The generation is done using a pseudo-random number generator (PRNG) with the seed is equal K2. The number of elements in each of the two pseudo-random sequences must be equal to the number of mid-band elements of the DCT-transformed blocks. The number of iteration k1 and the seed k2 can be considered together as the secret key.

5. Embed the scrambled watermark in the mid frequency DCT coefficients generated in step (2). Let X be the mid frequency coefficients of the DCT-transformed block, then embedding is done as follows:

   - If the watermark bit is 0 then: $X' = X + \alpha * \mathrm{PN}_0$,
   - If the watermark bit is 1 then: $X' = X + \alpha * PN_1$

6. Perform inverse DCT on each block after its mid frequency coefficients have been modified to embed the watermark bits. This step will generate the modified approximation sub-band $\widehat{LL}$.

7. Perform the inverse DWT on modified approximation sub-band $\widehat{LL}$ including the original details coefficient sets HL, LH and HH, to produce the watermarked host image.

The combined DWT-DCT algorithm is a blind watermarking algorithm and thus the watermark could be extracted without resorting to the original host image. The extraction process is given as follows:

1. Apply DWT to decompose the watermarked image into four non-overlapping sub-bands:LL, HL, LH, and HH.

2. Divide the sub-band LL into 8x8 blocks, and apply DCT on each block.

3. Re-generate the two pseudo-random sequences PN_0 and PN_1 using the PRNG and the same seed k2.

4. Extract the mid frequency coefficients of each DCT-transformed block, and calculate the correlation between the mid frequency coefficients and the pseudo random sequences PN_0 and PN_1: if the correlation with the PN_0 is higher than the correlation with PN_1, then the extracted watermark bit is considered as 0, otherwise the extracted watermark bit is considered as 1.

5. Reconstruct the scrambled watermark using the extracted watermark bits, and then recover the watermark with the inverse Arnold transform from the scrambled watermark using k1.

In addition of being blind, Feng et al. method has also proved an acceptable robustness against various attacks using common image processing such Gaussian noise, median filtering, JPEG compression, resizing and cropping [**Feng2010**]. For these reasons, this method is used in chapter 3 as a comparison method (along with other method) to assess the efficiency of our approach.

3.8 Conclusion

The DWT and DCT are the two most popular transform domains for digital image watermarking. The DWT is favored because it simultaneously explores spatial and spectral properties of the image. By contrast, the DCT holds the advantage of excellent energy compaction for highly correlated image data. Despite the wide variety of techniques proposed, all of them share the

same basic property: they keep track of the spatial characterization of the host signal, while at the same time exploiting the richness of the frequency interpretation.

In an attempt to contribute to a better understanding of the efficiency of combining more then one transform domain in image watermarking, we propose in the next chapter, two new image watermarking methods; one is based on the DCT and the other on the combination of the DWT and the DCT domains. Comparison between the two methods and discussions are also given.

Chapter 4

Proposed DWT and DCT image watermarking method

4.1 Introduction

This chapter deals with the design and analysis of two blind algorithms using DCT and DWT transforms for watermarking gray scale images. The first algorithm uses only the 1-DCT transform of two sub-vectors to embed the bits of the watermark sequence in a differential manner. These sub-vector are obtained by sub-sampling one vector derived from the zig-zag scanning of the image pixels. During the extraction stage, the simple difference between the corresponding sub-vectors of the watermarked image, gives directly the embedded watermark sequence. In the second algorithm, the whole process of the first algorithm is now applied on the approximation coefficients of the DWT transform of the host image. The results of comparison between the combined DCT-only, the DWT-DCT and other methods, justify the combining of DWT and DCT domains.

4.2 The embedding process

The proposed watermark embedding scheme is shown in Figure 4.1.

Let I denotes the gray-scale square image of size MxM to be watermarked by the bipolar $\{-1,1\}$ binary sequence W of size L, the embedding process can be described as follows:

- **Step 1:** Perform the 1-level DWT of the input image I. This produces the approximation coefficients matrix (LL sub-band) and a set of detail coefficients (HL, LH and HH sub-bands).

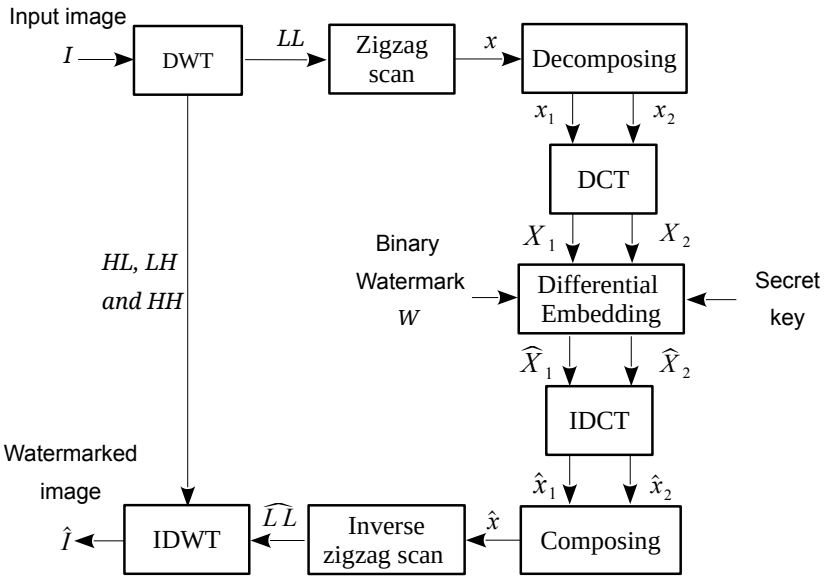


Figure 4.1: The Embedding procedure of the DWT-DCT method.

- **Step 2:** Perform zigzag scanning [**Bhatnagar2012**] to convert the matrix LL into a vector of approximation coefficients $x(n)$, $n = 1,...,N$, where $N = M.M/4$. Since adjacent pixels are highly correlated in real images and the first-level LL sub-band represents a close approximation of the original image; the zigzag scanning of the LL matrix helps to cluster high correlated approximation coefficients in the vector $x(n)$.

- **Step 3:** Decompose the vector of approximation coefficients x into two (correlated) sub-vectors x_1 and x_2 using the following sub-sampling operations:

$$x_1(k) = x(2k) \tag{4.1}$$

$$x_2(k) = x(2k-1) \tag{4.2}$$

where $k = 1,...,N/2$.

- **Step 4:** Perform DCT on x_1 and x_2 to produce their DCT-transformed versions X_1 and X_2 (Figure 4.2):

$$X_1 = dct(x_1) \tag{4.3}$$

$$X_2 = dct(x_2) \tag{4.4}$$

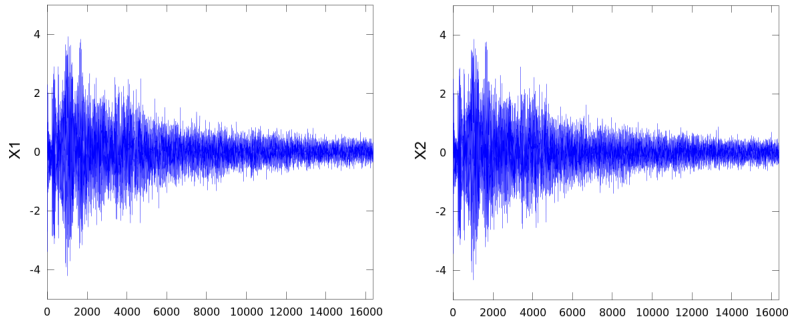


Figure 4.2: The first half of DCT-transformed sub-vectors X_1 and X_2 (without the DC component) for the image of Peppers.

- **Step 5:** Insert the watermark sequence bits $W(i)$ for $i = 0, 2, ..., L-1$, in the transformed sub-vectors X_1 and X_2 using a differential embedding technique. This will produces two transformed and modified (by watermarking) sub-vectors $\hat{X}_1$ and $\hat{X}_2$ as follows:

$$\hat{X}_1(i') = \frac{1}{2}\left[X_1\left(i'\right) + X_2\left(i'\right)\right] + \alpha W\left(i\right) \tag{4.5}$$

$$\hat{X}_2(i') = \frac{1}{2}\left[X_1\left(i'\right) + X_2\left(i'\right)\right] - \alpha W\left(i\right) \tag{4.6}$$

where α is the gain factor and i' are the random locations within the high energy band of X_1 and X_2 in which the watermark bits are embedded (Figure 4.3). These locations are the elements of a vector r which can be generated using a random permutation function:

$$i' = r(i) \tag{4.7}$$

$$r = RandPerm(S, a, b) \tag{4.8}$$

where S is the seed of the associated pseudo random number generator (PNRG), a and b are the starting and the ending locations of the high energy band used to insert the watermark (Figure 4.3). Therefore, the user's secret key is $key = (S, a, b)$, which prevent the watermark from tempering or unauthorized access by attackers.

Step 6: Perform the inverse DCT on $\hat{X}_1$ and $\hat{X}_2$:

$$\hat{x}_1 = idct(\hat{X}_1) \tag{4.9}$$

$$\hat{x}_2 = idct(\hat{X}_2) \tag{4.10}$$

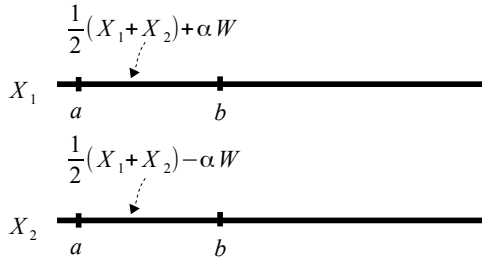


Figure 4.3: The differential embedding of the watermark in the high energy band of the transformed sub-vectors X_1 and X_2.

Step 7: Combine the two modified sub-vectors $\hat{x}_1$ and $\hat{x}_2$ using the opposite operation in (4.1) and (4.2) in order to produce the modified vector of approximation coefficients $\hat{x}$:

$$\hat{x}(2k) = \hat{x}_1(k) \tag{4.11}$$

$$\hat{x}(2k-1) = \hat{x}_2(k) \tag{4.12}$$

for $k = 1,...,N/2$.

Step 8: Convert the modified vector $\hat{x}$ into the matrix of a modified approximation coefficients $\widehat{LL}$ using the inverse of the zigzag scan operation used in step 2 .

Step 9: Construct the watermarked image $\hat{I}$ by performing the inverse wavelet transform of the modified approximation coefficients $\widehat{LL}$ and the sets of original detail coefficients (HL, LH and HH sub-bands).

Note 1: The parameters a and b should be chosen to satisfy the following conditions:

- $a > 0$: The DC-components of the transformed sub-vectors X_1 and X_2 must remain unchanged in order to preserve the quality of the watermarked image;
- $b - a \geq L$: The insertion band have to be wide enough to insert all the watermark's bits;
- $b \leq \frac{N}{2}$: The watermarking is done in the hight energy band of X_1 and X_2 in order to guarantee the robustness of the method.

Note 2: While the normal differential embedding would be as follows: $\hat{X}_1 = X_1 + \alpha W$ and $\hat{X}_2 = X_2 - \alpha W$ (by omitting the insertion locations), the fact that the transformed sub-vectors X_1 and X_2 are highly correlated (as shown in Figure 4.4) allow as to assume in Eq. (5) and (6), that $X_1 \approx X_2 \approx \frac{1}{2}[X_1 + X_2]$. This will ensure that X_1 and X_2 are equally contributing in the new modified ones $\hat{X}_1$ and $\hat{X}_2$ so that the resulting distortion on the watermarked image will be minimal. Also, it is obvious that the difference between $\hat{X}_1$ and $\hat{X}_2$ will give an amplified (by 2)

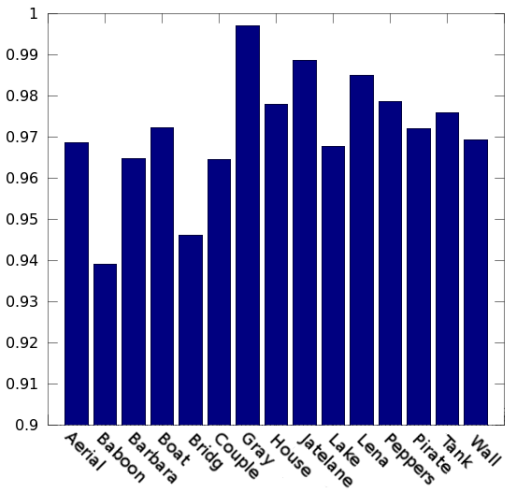


Figure 4.4: Correlation between X_1 and X_2 for 15 test images.

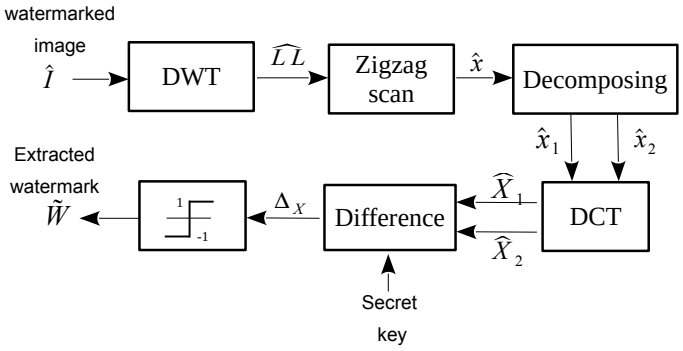


Figure 4.5: The Extraction procedure of the DWT-DCT method.

amount of the inserted watermark sequence (αW) which is the key feature of this differential embedding technique.

4.3 The extraction process

The watermark extraction process follows the same steps as the embedding process until Step 5 where the extraction is taking place as shown in Figure 4.5.

If the input image is the watermarked one, and by analogy with the embedding process, the step 4 of the extraction process will give the two sub-vectors $\hat{X}_1$ and $\hat{X}_2$ in Equ. 5 and 6 respectively. Consequently, the difference between them has a proportional relationship with the watermark sequence W:

$$\triangle_X(i) = \hat{X}_1(i') - \hat{X}_2(i') = 2\alpha W(i) \tag{4.13}$$

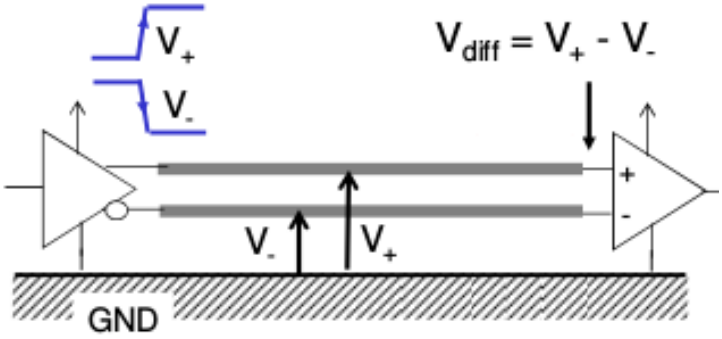


Figure 4.6: A differential-signaling system [**Johnson2003**].

With $i = 1, ..., L$, and i' are the random locations where the watermark bits are embedded. These locations are determined simply by recreating the vector r (Equ. 7) using the user-selected secret key and the random permutation function (Equ. 8). Finally, and since the difference $\triangle_X(i)$ might differ from +1/-1 values, we apply a hard limitation function on it in order to recover the original bits of the watermark:

$$\tilde{W}(i) = \begin{cases} 1 & if \quad \triangle_X(i) \geq 0 \\ -1 & otherwise \end{cases} \tag{4.14}$$

If the watermarked image has been attacked, the proposed method is able to extract the watermark and the quality of extraction is closely depend on the severity of the attack as shown in the experiments.

Notice that no threshold setting is needed at the extraction stage, which represents a great advantage compared with a lot of schemes in literature [**Kang2003**; **Zhao2004**]. Also, this watermarking approach is analogous to the technique of *Differential-Signaling*, (Figure 4.6) a method of transmitting information electrically by means of two complementary signals [**Johnson2003**]. At the end of the connection, the receiver reads the difference between the two signals to recover the original information.

4.4 The DCT-only method

To justify the utility of combining DWT and DCT domains, we propose here a reduced version of our method based only on the DCT domain. Since we applied the zig-zag scanning on the matrix of approximation coefficients of DWT (Section 2.1) to obtain the vector x, we choose here to apply it on the whole image (without using DWT). The flowchart of the embedding process of the DCT-only method is given in Figure 4.6.

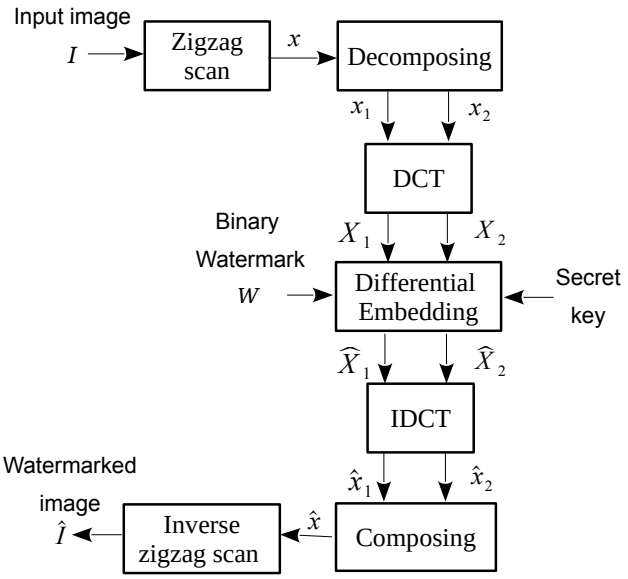


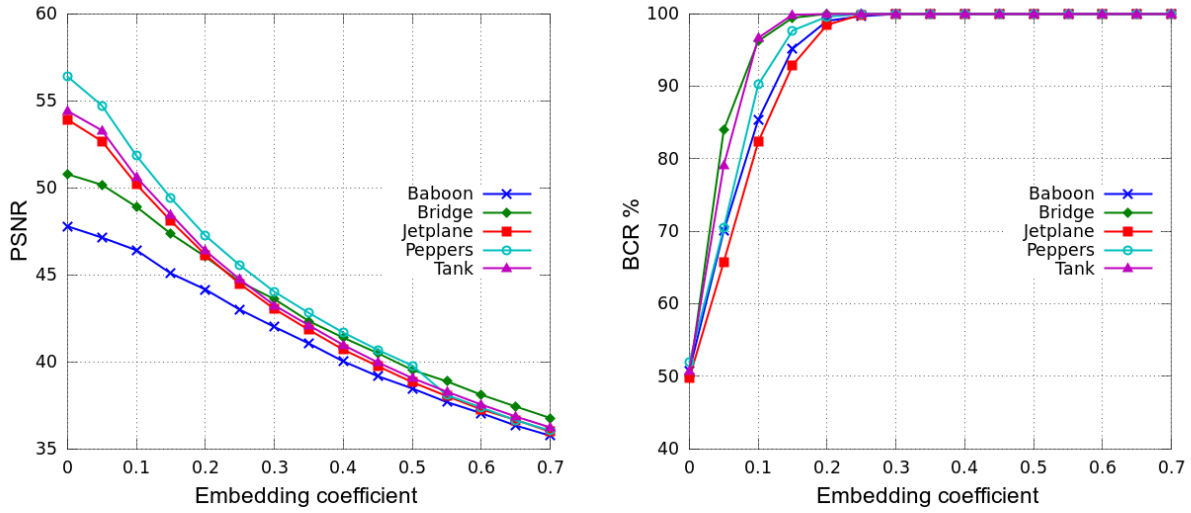
Figure 4.7: The Embedding procedure of the DCT-only method.

In this process, the operations are the same as described in the embedding process of the DWT-DCT method (Section 2.1). Notice that all the vectors in Figure 4.6 are four times the size of the corresponding vectors in the DWT-DCT method. The extraction process will be the same as in Figure 4.5 except that the DWT step is not needed in this case.

4.5 EXPERIMENT RESULTS

In this section, several experiments are conducted to evaluate the performance of the proposed watermarking scheme. The size of original cover images (Baboon, Bridge, Jetplane, Peppers and Tank) is 512×512 pixels [1]. The watermark is a pseudo-random binary sequence of size 256 bits, which is a usual payload [**Yaghmaee2010**]. Haar wavelet transform [**Strang1997**]is used in the wavelet decomposition of the image in the DWT/DCT method. To evaluate the quality of the watermarked image, the peak signal to noise ratio (PSNR) is used (Section 2.5). To evaluate the quality of extracted watermark, the bit-correct ratio (BCR) is adopted to measure the similarity between the original watermark W and the extracted watermark $\tilde{W}$ (Section 2.5).

[1] All test images are obtained from the USC-SIPI Image Database: http://sipi.usc.edu/database/



(a) (b)

Figure 4.8: PSNR of watermarked images versus the gain factor α (a). BCR of the extracted watermark versus the gain factor α after JPEG lossy compression (Q=40) (b).

4.5.1 Gain factor selection

In order to select the suitable values of the gain factor α in Eq. (5) and (6) that fulfill both the invisibility and the robustness requirements of the watermarking, we plot with respect to α the PSNR of the watermarked images (Figure 4.7-a) and the BCR of the extracted watermarks (Figure 4.7-b) after a standard image attack (JPEG lossy with quality factor equals to 40).

It is apparent from Figure 4.7 that higher α values make lower PSNR of the watermarked images, but the similarity ($BCR\%$) of original watermark and the extracted watermark gets better for higher values of α. The best trade-off between visual quality and watermark robustness is achieved for the values of α in range from 0.2 to 0.3 where the PSNR values are greater than 40 dB and the BCR values are almost 100 for all test images. In the rest of our experiments we will set α= 0.3 as the default gain factor value.

Figure 4.8 shows the perceptual difference between the original test images and their watermarked versions and the corresponding PSNR measures.

From Figure 4.9 we can see that PSNR of 15 watermarked images is greater than 42 dB which ensures the invisibility requirement. Notice that the watermark is extracted from all test images with no error ($BCR = 100\%$).

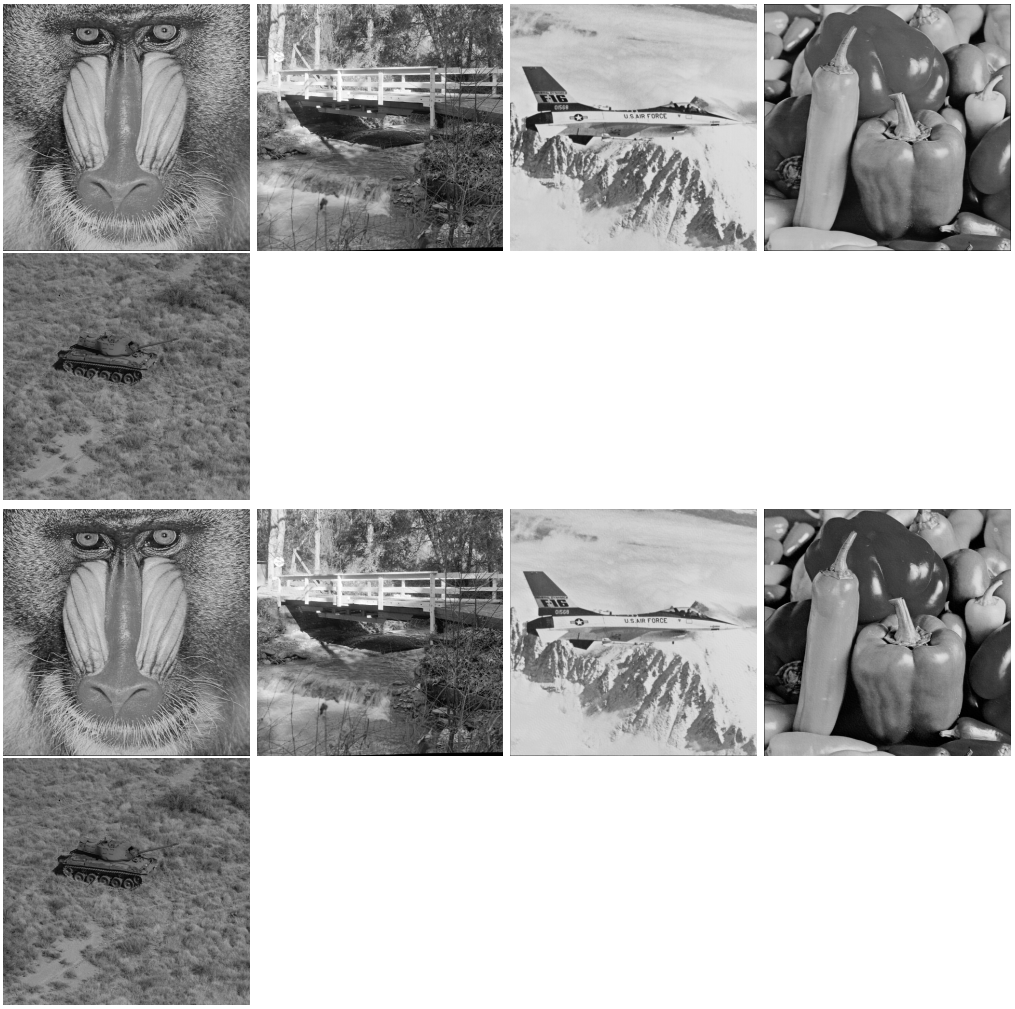


Figure 4.9: The original images of Baboon, Bridge, Jetplane, Peppers and Tank (Top), and the their watermarked versions (Bottom).

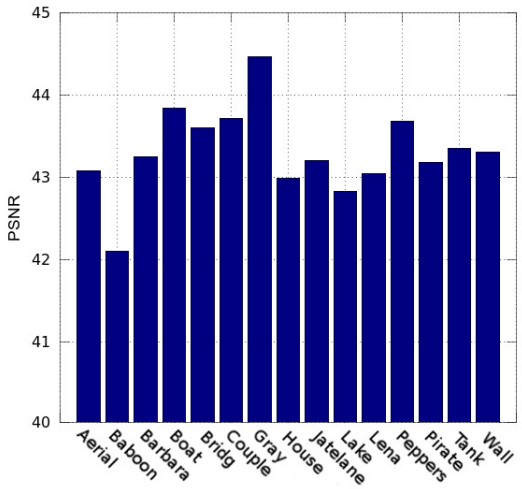
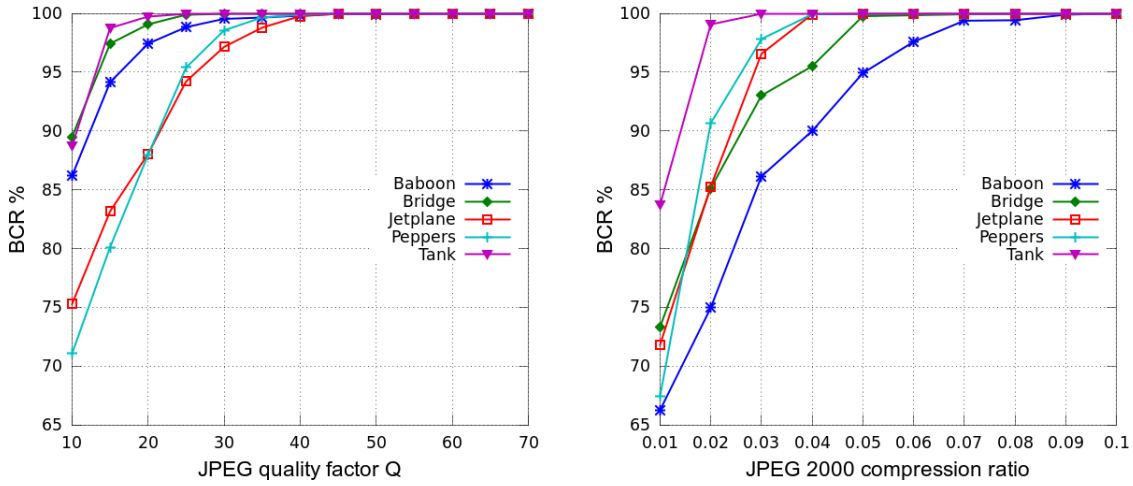


Figure 4.10: PSNR of 15 watermarked images.



(a) (b)

Figure 4.11: Robustness against compression attacks. (a): JPEG lossy. (b): JPEG2000.

4.5.2 Robustness tests

4.5.2.1 Robustness against image compression

The BCR values of the extracted watermarks under JPEG lossy and JPEG2000 compression attacks are shown in Figure 4.10. For JPEG lossy compression, the quality factor (Q) is varied from 10 to 70, whereas for JPEG2000 attacks, the compression ratio r [**Qadir2010**] is varied from 0.01 to 0.1.

We can see from Figure 4.10-a that the watermark is completely recovered under high strength JPEG lossy attacks ($BCR = 100$ for $Q \geq 40$) for all test images.

The proposed method is also robust for the practical JPEG2000 compression range of levels, i.e. $r \geq 0.04$ (Figure 4.10-b) except for the textured images of Baboon and Bridge where it exhibits a lower robustness. This because, JPEG2000 compression does not necessarily provide an image of good quality in texture features [**Roimela2008**].

4.5.2.2 Robustness against image processing attacks

In the following, we evaluate the proposed method against noise addition, low pass filtering, image enhancement, etc. Table 4.1 shows the BCR values of the extracted watermarks under noise addition attacks. For the three types of noises (Gaussian, Salt&pepper and Speckle) we can observe that the proposed method is fairly robust against noises with medium variances,

whereas for high variance noises the method presents acceptable performance since the BCR values is greater than 95% for the majority of experiments.

Attack	Image				
	Baboon	Bridge	Jetplane	Peppers	Tank
Gaussian noise (var=0.005)	98.6	98.9	98.2	99.3	99.1
Gaussian noise (var=0.01)	94.3	95.0	96.2	95.7	96.6
Salt & pepper noise (var=0.01)	99.6	99.7	99.1	99.7	99.6
Salt & pepper noise (var=0.02)	97.8	97.9	98.2	98.1	99.1
Speckle noise (var=0.01)	99.5	99.8	98.1	100	99.6
Speckle noise (var=0.02)	98.3	98.9	96.0	99.4	99.5

Table 4.1: BCR(%) results of the suggested blind watermarking technique under noise addition attacks

Tables 4.2 and 4.3 demonstrate that the proposed method is robust against low-pass filtering, Bit-plane removal, Gamma correction, Histogram equalization, and Laplacian sharpening attacks. Notice that in bit plane removal attack, the least significant bits of the watermarked image are replaced with zeros. All these experiments show that the proposed method is robust against common image processing attacks.

Attack	Image				
	Baboon	Bridge	Jetplane	Peppers	Tank
Average filter (3x3)	99.3	99.6	99.4	99.7	99.9
Average filter (5x5)	53.6	52.5	54.5	54.4	59.5
Gaussian filter (5x5) var=1.5	90.6	92.4	95.0	90.3	99.1
Gaussian filter (5x5) var=1	99.3	99.7	98.8	99.4	99.8
Median filter (3x3)	97.0	99.1	98.4	99.5	99.7
Wiener filter (3x3)	99.6	99.9	99.9	100	100

Table 4.2: BCR(%) results of the suggested blind watermarking technique under low-pass filtering attacks.

Figure 4.12: Attacked images of Baboon, Bridge, Jetplane, Peppers and Tank with LSB removal (6 bits), JPEG lossy (Q = 20), Gamma correction (Gamma=3), Gaussian noise (var=0.007), and Gaussian low pass filter (5x5). The BCR value is 100% for all these attacks.

Attack	Image				
	Baboon	Bridge	Jetplane	Peppers	Tank
Bit-plane removal (5 bits)	99.9	99.7	97.5	99.9	99.8
Bit-plane removal (6 bits)	99.1	91.5	96.2	98.7	98.5
Gamma correction (0.5)	100	100	100	100	100
Gamma correction (1.5)	100	100	100	100	100
Histogram equalization	100	100	100	100	100
Laplacian sharpening	100	100	100	100	100

Table 4.3: BCR(%) results of the suggested blind watermarking technique under other image processing attacks.

Figure 4.11 shows the visual impact of some attacks on different images. The watermark is extracted with BCR value greater than 99% which confirms the preceding results.

4.5.2.3 Robustness against geometrical attacks

The next experiments shows the robustness against some geometrical attacks on the test images.

Table 4.4 and Figure 4.12 demonstrate that the proposed method is relatively robust against geometrical attacks. In particular, it performs well for cropping and resizing attacks, but exhibits weak robustness against rotation attacks.

Figure 4.13: Robustness of the DWT-DCT technique against different type of cropping. The BCR values is 100\% for these attacks.

Attack	Image				
	Baboon	Bridge	Jetplane	Peppers	Tank
Resizing ($512 \rightarrow 256 \rightarrow 512$)	98.2	99.8	99.8	100	100
Resizing ($512 \rightarrow 200 \rightarrow 512$)	80.4	86.1	97.1	98.2	97.5
Rotation (0.25°)	80.9	90.3	96.4	99.1	98.4
Rotation (0.5°)	58.8	55.5	60.7	60.5	61.4
Surrounding crop (15%)	97.8	98.9	98.2	99.8	99.7
Surrounding crop (25%)	95.4	98.1	97.5	99.6	99.3

Table 4.4: BCR(%) results of the suggested blind watermarking technique under geometrical attacks.

4.5.2.4 Robustness against watermark suppression attack

In these experiments we suppose that the an informed attacker have partial knowledge of the embedding process and he tries to perform a successful attack that produces a smaller amount of perceptible distortion compared to its blind counterparts. In particular, we suppose that the attacker tries to obliterate the inserted watermark by erasing a portion of the two carrying sub-vectors $\hat{X}_1$ and $\hat{X}_2$ (Eq. (5) and (6)). So the attacker puts the vectors $X_1(n) = 0$ and $X_2(n)$ to 0 for $n = 1, 2, ..., K$. ($n \neq 0$ in order to keep the DC value intact). The results of attacked images of Peppers and the corresponding BCR values for different value of k are given in Figure 4.13. We can see that the watermark (or a portion of it) can be successfully extracted for this type of attacks. However, once the suppression of the watermark exceeds certain level ($K > 512$), the image become no longer usable because of the high perceptible distortions.

Figure 4.14: Robustness against watermark suppression attack for the image of Peppers. From left to right : the values of K are 128, 256, 512 and 1024 and the corresponding values of BCR are 100, 99.6, 98.8 and 91.8 receptively.

	DCT-only	DWT/DCT
Embedding	0.20	0.27
Extracting	0.20	0.17

Table 4.5: Comparative execution times (in seconds) for the proposed methods.

4.5.3 Time execution performance

The computational cost of the DCT-based and the DWT/DCT methods is very acceptable as shown in Table 4.5. This is because in the two methods, the DCT transform is operating on 1-d victors instead on 2-D matrices as opposed to almost all DCT-based watermarking methods which are block-based techniques.

Note that the simulation of the watermarking techniques is performed using Octave 3.2 (Appendix A) under 32bits Linux distro (ubuntu 12.04) running on Intel Core-i3 (3.07GHz) and 2GB RAM machine. Notice that we have used mex-compiled function to speed up the zigzag operations for the two methods.

4.5.4 Comparison with other methods

In this subsection, we conduct several experiments to compare the performance of the proposed DWT-DCT method with two other blind watermarking approaches in [**Feng2010**] and [**Lin2008**]) and also with the reduced DCT-only method.

The approach in [**Lin2008**] proposed a block-based significant difference quantization watermarking. Every seven wavelet coefficients in one sub-band are grouped into a block and the watermark bit is embedded into a block by quantizing the difference between two maximum

Attack	Method			
	Feng	Lin	DCT-only	DWT-DCT
JPEG 2000 (R=0.02)	79.8	97.2	86.8	90.2
JPEG 2000 (R=0.03)	92.2	98.8	100	99.6
JPEG 2000 (R=0.04)	98.4	100	100	100
JPEG lossy (Q=40)	89.2	100	74.9	100
JPEG lossy (Q=30)	78.4	98.4	68.7	99.4
JPEG lossy (Q=20)	69.8	94.5	61.4	88.8

Table 4.6: Comparison of robustness (BCR) against image compression attacks between Fang's method [**Feng2010**], Lin's method [**Lin2008**], the DCT-only and the DWT-DCT methods.

Attacks	Method			
	Feng	Lin	DCT-only	DWT-DCT
Bit-plane removal (5 bits)	96.3	85.9	100	100
Gamma correction (3)	99.2	76.1	100	100
Gaussian noise (0.01)	87.5	79.5	96.3	94.9
Histogram equalization	99.3	94.5	100	100
Laplacian sharpening	100	91.4	100	100
Median filter (3x3)	97.1	99.2	41.6	100
Gaussian filter (5x5)	98.8	94.5	83.4	100
Salt & pepper noise (0.02)	93.3	85.9	99.1	98.3

Table 4.7: Comparison of robustness (BCR) against image processing attacks between Fang's method [**Feng2010**], Lin's method [**Lin2008**], the DCT-only and the DWT-DCT methods.

wavelet coefficients. Notice that the embedding parameters in each method are adjusted to produce a watermarked image (Lena) of PSNR equal to 44 dB. The results of comparison are listed in Tables 4.6, 4.7 and 4.8.

From the above results, we notice the following:

- The proposed DWT-DCT method outperforms the reduced DCT-only method for JPEG compression, low-pass filtering, resizing and rotation attacks. For the rest of attacks, both techniques perform equally well. Consequently the combination of the two transforms (DWT and DCT) is more practically helpful than the use of one domain only (DCT) especially if the watermarked images are intended to undergo these types of attacks.

Attacks	Method			
	Feng	Lin	DCT-only	DWT-DCT
Resizing($512 \rightarrow 200 \rightarrow 512$)	98.9	94.1	65.6	100
Rotation (0.25°)	90.6	88.2	65.5	99.4
Top left quarter crop	88.3	85.1	100	99.1
Surrounding crop (25%)	82.2	82.0	99.7	99.2

Table 4.8: Comparison of robustness (BCR) against geometrical attacks between Fang's method [**Feng2010**], Lin's method [**Lin2008**], the DCT-only and the DWT-DCT methods.

- For JPEG compression, only the method in [**Lin2008**] performs slightly better than the proposed DWT-DCT method. This is because the fact that wavelet quantization techniques are generally robust against image compression attacks [**Lin2008**].

- For the rest of attacks, the proposed DWT-DCT method is more robust than the two methods in [**Feng2010**] and[**Lin2008**] especially for Bit-plane removal, Gamma correction, noise addition and for all geometrical attacks.

From the previous results we may conclude that, overall, the proposed method has a better performance than the compared watermarking schemes and that the combination of the DWT and DCT domains is more advantageous than the use of only one frequency domain.

4.6 Conclusion

In this chapter, a robust, yet simple watermarking scheme based on the combination of DWT and DCT domains is presented. In the embedding process, a differential technique is performed on two transformed sub-vectors so that the extraction of the watermark is achieved using only the difference of the corresponding watermarked sub-vectors.

Overall, the experimental results demonstrate that our scheme provides excellent robustness against a multiple image attacks such as bit-plan removal, cropping, JPEG compression, histogram equalization, low-pass filtering, and noise adding attacks. Besides, the quality of the watermarked image is satisfactory in term of imperceptibility as the PSNR per watermarked image is over 42 dB.

We have also investigated the utility of the combination of the DWT and DCT transforms through the proposition of a relaxed version of our method based only on the DCT transform. In comparison, the DWT-DCT method is more robust than the DCT-only method for a set of attacks such as JPEG compression and low-pass filtering. The results of experiments have

showed also that the proposed (DWT-DCT) method has stronger robustness in comparison with two existing watermarking schemes.

In the next chapter, we will extend the proposed approach to video watermarking domain. As the embedding and the extracting processes are low complexity, and doesn't require any specific features of the input image, the extension to video watermarking will be straightforward.

Chapter 5

Proposed Video Watermarking Method

5.1 Introduction

In this chapter we will extend the proposed image DCT-based watermarking techniques (chapter 3) to color video watermarking. This approach uses the differential embedding technique to insert the bits of watermark into the frames of the video segment so that the extraction process is blind and straightforward. To further ensure the security of the method, a binary image watermark is scrambled using Arnold transform before embedded into the video segment. Also, a color space transformation from RGB to YUV is performed in order to deal with the color nature of the video segments. The proposed approach exhibits good robustness against a wide range of attacks such as video compression, cropping, Gaussian filtering, and noise adding.

5.2 Video watermarking

Video watermarking is similar to image watermarking where the video can be divided into a number of frames with all the frames being watermarked individually [**Agrawal2016**]. Usually, a person viewing the video cannot find the difference between the original video and the watermarked video, but a watermark extraction process can identify the watermark and obtain the embedded watermark information [**Ramkumar2014**]. Recently, transform-based video watermarking schemes have attracted a lot of attention especially those based on DCT and DWT [**Agarwal2014**; **Preda2011**; **Ramkumar2014**].

Preda et. al. [**Preda2011**] proposed the novel digital watermarking method for video based on a multi-resolution wavelet decomposition. In their scheme, they used the binary image

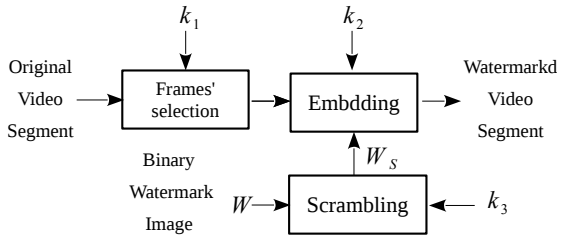


Figure 5.1: The general video embedding process.

watermark i.e. embedded into the wavelet coefficients of the LH, HL and HH sub-bands of the second wavelet decomposition level by quantization.

Wang et. al. [**Wang2012**] have developed a real-time robust video watermarking scheme for MPEG-2 compressed video. In their proposed scheme they embed the watermark into the histogram bins calculated from the low■ frequency sub band of the DWT domain. Their scheme is specially robust against geometric distortions such as rotation, cropping, scaling, and frame dropping.

Farfoura et al. [**Farfoura2016**] proposed a blind semi-fragile watermarking scheme for authenticating the integrity of H.264/AVC videos. The Watermark embedding is performed by flipping the signs of nonzero DCT coefficients of candidate pairs of certain blocks of the video frames which are selected based on a spatial analysis to ensure imperceptibility and robustness. The experimental results show that their scheme has a high resilience against content-preserving attacks while it shows high sensitivity against content-changing attacks.

5.3 The general embedding and extracting process

The proposed video watermarking comprises different modules such as video pre-processing, watermark pre-processing (scrambling), watermark embedding, and watermark extraction. The general embedding process is shown in Figure 5.1 and it comprises the following steps:

Step 1: Get an uncompressed video segment and convert it into frames.

Step 2: Select a random set of frames to embed the watermark using a secret key ($K1$). The number of the selected frames must be equal to the number of columns of the binary image watermark (W).

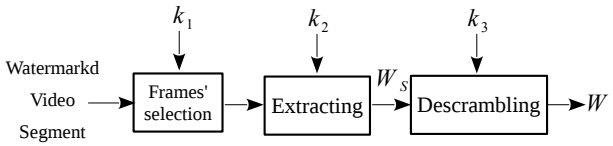


Figure 5.2: The general video extracting process.

Step 3: Scramble the binary watermark image using Arnold transform and the secret key ($K2$). Given an $H \times H$ square image, one of its discretized versions [**Khalili2013**] is defined as follows:

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = \begin{pmatrix} 1 & a \\ b & ab+1 \end{pmatrix} \begin{pmatrix} x_i \\ y_i \end{pmatrix} \bmod(H) \tag{5.1}$$

where a and b are parameters that can serve as the secret key if the function is used for encryption purposes. So the secret key ($K3$) is composed of three parameters : a,b and the number of iteration i.

Step 4: Embed each column of the scrambled watermark image in the Y channel of the YUV space representation of the selected frame (Step 2) using the differential embedding method shown in Figure 5.3.

Step 5: Concatenate all frames (watermarked and not) into one watermarked video segment.

On the other hand, the extraction process of the binary watermark image from the watermarked (and possibly attacked) video segment is inversely analogous to the embedding process and is given as follows:

Step 1: Using the secret key ($K1$), select the set of frames that have been used in the watermark embedding phase.

Step 2: Extract each column of the scrambled watermark image from the selected frame using the extracting process shown in Figure 5.4. The concatenation of all extracted columns constitutes the scrambled watermark W_s.

Step 3: De-scramble the scrambled watermark image W_s using the inverse Arnold transform and the secret key ($K3$) to produce the original binary watermark image W.

5.4 Proposed DCT-based video watermarking method

The basic process of this frame embedding technique is given in Figure 5.3 and it comprises the following sub-steps:

1. Convert the input frame from RGB to YUV color space and get the intensity Y channel matrix of size $M \times N$. The convertion from RGB to UYV color spaces is given by [**Singh2013**]:

$$\begin{cases} Y = 0.299R + 0.587G + 0.114B \\ U = -0.147R\text{-}0.289G + 0.436B \\ V = 0.615R - 0.515G\text{-}0.100B \end{cases} \tag{5.2}$$

2. Perform zigzag scanning on the Y channel matrix to convert it into one vector x,

3. Decompose the vector x into two sub-vectors x_1 and x_2 using sub-sampling:

$$x_1(k) = x(2k) \tag{5.3}$$

$$x_2(k) = x(2k-1) \tag{5.4}$$

   where $k = 1, ..., M * N$,

4. Perform DCT on x_1 and x_2 to produce their DCT-transformed versions $X_1$and X_2,

5. Insert the vector of the watermark bits w_s (which is one column of W_s) in a random locations (chosen using a secret key $K3$) of the the transformed sub-vectors $X_1$and X_2according the following equations [**Benoraira2015**]:

$$\hat{X}_1 = \frac{1}{2}(X_1 + X_2) + \alpha w_s \tag{5.5}$$

$$\hat{X}_2 = \frac{1}{2}(X_1 + X_2) - \alpha w_s \tag{5.6}$$

6. Perform the inverse DCT on $\hat{X}_1$ and $\hat{X}_2$ to obtain a modified sub-vectors $\hat{x}_1$and $\hat{x}_2$,

7. Combine the two modified sub-vectors $\hat{x}_1$and $\hat{x}_2$ using the up-sampling operation given in Equ. 2 and 3 in order to produce the modified (watermarked) vector $\hat{x}$.

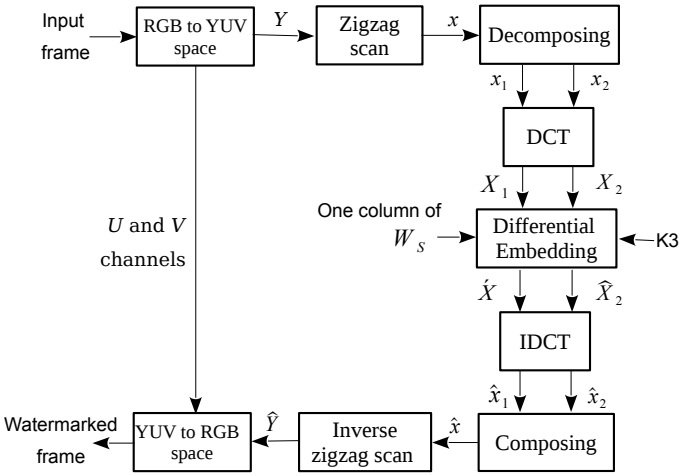


Figure 5.3: The embedding process for video frames.

8. Convert the new vector $\hat{x}$ into the matrix of a modified intensity channel $\hat{Y}$ using the inverse of the zigzag scan operation.

9. Construct the watermarked frame from the watermarked intensity channel $\hat{Y}$ and the two original color channels (U,V) and then convert $\hat{Y}$ to RGB color space using the formula [**Singh2013**]:

$$\begin{cases} R = Y + 1.40V \\ G = Y - 0.395U\text{-}0.581V \\ B = Y + 2.032U \end{cases} \tag{5.7}$$

Notice that the difference between $\hat{X}_1$ and $\hat{X}_2$ in (4) and (5) has a proportional relationship with the watermark vector ($\triangle_X = \hat{X}_1 - \hat{X}_2 = 2\alpha w_s$). Therefore, if the input frame is a watermarked one, and by analogy with the embedding process, the extraction process (Figure 5.4) will produce one watermark vector w_s as explained in Section 3.2.

If the watermarked video segment has been attacked, this video watermarking method is able to extract the watermark and the quality of extraction is closely depend on the severity of the attack as shown in the experiments.

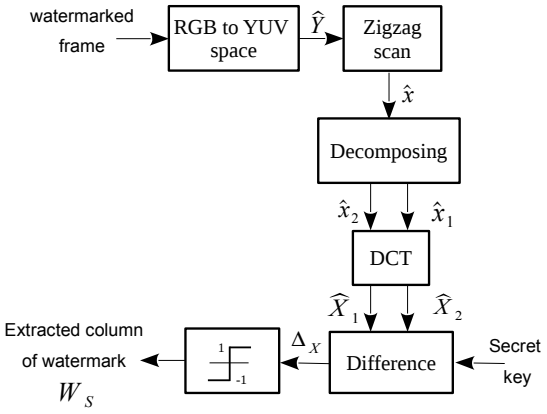


Figure 5.4: The Extraction process of the DCT method.

Figure 5.5: The binary watermark image and its scrambled version using Arnold transform.

5.5 Experimental results

The proposed algorithm is evaluated for five AVI format uncompressed video sequences of size 300 frames of 352x288: 'City', 'Coastguard', 'Crew', 'Foreman', and 'Soccer', available from the Xiph.org database[1]. The chosen watermark is a binary image of size 100×100 shown in Figure 5.5. Also, we set α= 0.3 as the default gain factor value as suggested in Section 3.4.

Figure 4.6 shows original video frames as well as the watermarked ones for the five test video segments. It is obvious that the quality of the watermarked frames is preserved qualitatively and quantitatively.

[1]https://media.xiph.org/video/derf/

Figure 5.6: Original (Top) and watermarked (Bottom) frames. The average PSNR value is about 41dB for all video segments.

5.6 Robustness Test

5.6.1 Robustness against common attacks

We evaluated the robustness of our video watermarking method to noise addition, Gaussian filtering, Gamma correction, sharpening.

Attack	Video				
	city	coastguard	crew	foreman	soccer
Gaussian noise (var=0.01)	100	100	100	100	100
Gaussian noise (var=0.02)	99.89	99.87	99.80	99.70	99.87
Gaussian noise (var=0.03)	99.39	99.02	99.31	99.03	99.17
Salt & pepper noise (var=0.01)	100	100	100	100	100
Salt & pepper noise (var=0.02)	100	100	100	100	100
Salt & pepper noise (var=0.03)	100	100	100	100	100

Table 5.1: Robustness of the proposed video watermarking technique against noise adding attacks.

It's clear that the proposed DCT-based video watermarking is robust against all the above attacks even for high intensity ones like the case of Gaussian noise of high variance.

5.6.2 Robustness against compression attacks

To access the robustness against video compression, we use 'Motion JPEG', 'Motion JPEG2000', and 'H.264' compression standards. The plot of the results of robustness against these attacks are shown in Figures 4.8, 4.9 and 4.10 respectively. We can see clearly from these

Attack	Video				
	city	coastguard	crew	foreman	soccer
Gaussian filter (5x5) var=1	100	100	100	100	100
Gaussian filter (3x3) var=1	100	100	100	100	100
Gamma correction (2)	100	100	100	100	100
Laplacian sharpening	100	100	100	100	100
Surrounding crop (10%)	100	100	100	100	100
Surrounding crop (25 %)	100	100	100	100	100

Table 5.2: Robustness of the proposed video watermarking technique against different types of attacks.

results that the presented watermarking algorithm exhibits very good performance against motion JPEG and JPEG2000 compression standards. On the other hand, the algorithm shows lower performance against H.264 attacks due to the very high compression rate of this standard.

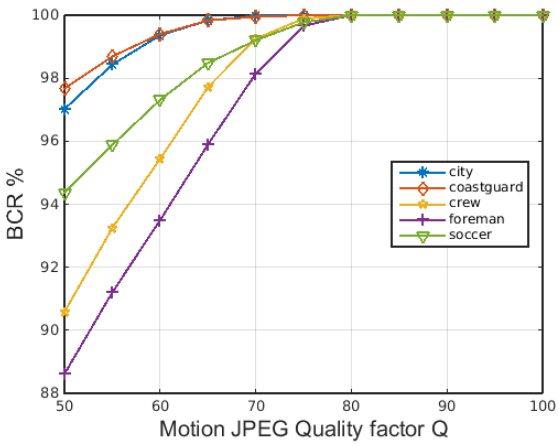


Figure 5.7: Robustness against Motion JPEG attack.

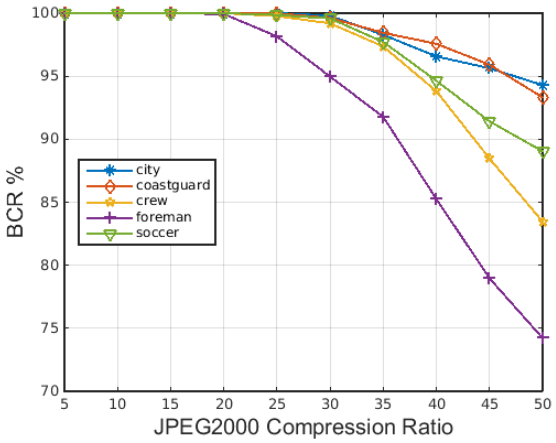


Figure 5.8: Robustness against Motion JPEG2000 attack.

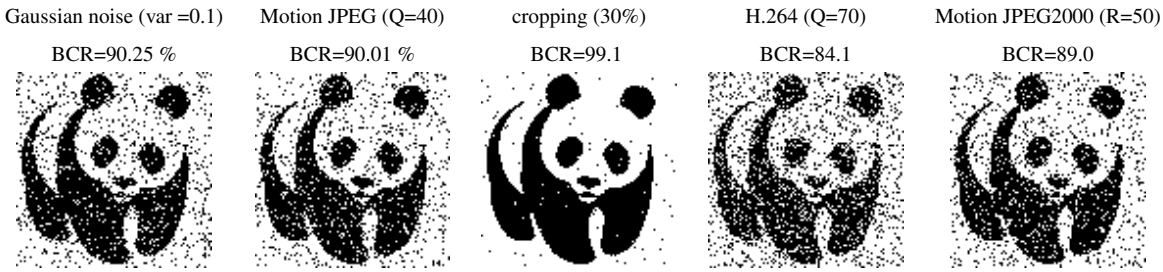


Figure 5.10: The extracted watermark under different types of attacks.

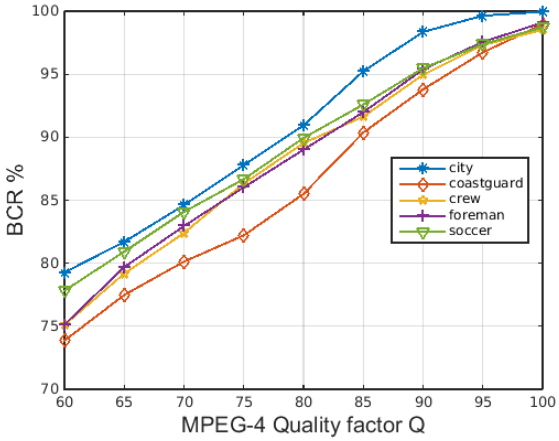


Figure 5.9: Robustness against H.264 compression attack.

Figure 5.10 shows the extracted visual watermarks after different types of high-strength attacks for the video segment of “soccer”.

5.7 Conclusion

In this chapter we have designed a robust and blind watermarking technique for uncompressed video using the DCT transform. The technique is based on the differential embedding of the bits of the watermark in DCT domain. The perceptual quality of the watermarked video segments and average PSNR values have been very good for all tested video segments. We have performed different attacks on the video frames and calculated the corresponding BCR values and have shown that the proposed technique fairly fulfills the requirements of watermarking.

Chapter 6

Biometric video watermarking using Raspberry Pi

6.1 Introduction

This chapter deals with the implementation of the proposed video watermarking technique using a real hardware device. The chosen hardware is the Raspberry Pi platform because its simplicity to setup, its availability and its educational aspect [**Severance2013**]. We have chosen to implement the DCT-only video watermarking method in order to increase the processing time of the system.

Instead of using an arbitrary binary watermark (as in chapter 4), we can effectively use a fingerprint image (of size200×200) issued from a fingerprint scanner to watermark the video segment. Therefore, the watermarking process will be much more secure because the fingerprint patterns are highly person-dependent [**Dutta2014**].

6.2 Biometric watermarking

In the ordinary methods of watermarking, the digital watermark is either generated from a pseudo-random number sequence, chaotic sequence or a binary image. In case of piracy dispute, it will be difficult to prove ownership of a digital watermark as an arbitrary sequence cannot be claimed for ownership, as it cannot be physically owned [**Dutta2014**]. A possible solution to this limitation, is to incorporate biometric features as the seed of the watermark (the user's key) or as the digital watermark itself [**Kannan2013**]. Watermarking of biometric data might also denotes a biometric image/template being watermarked for its authenticity [**Shaw2013**].

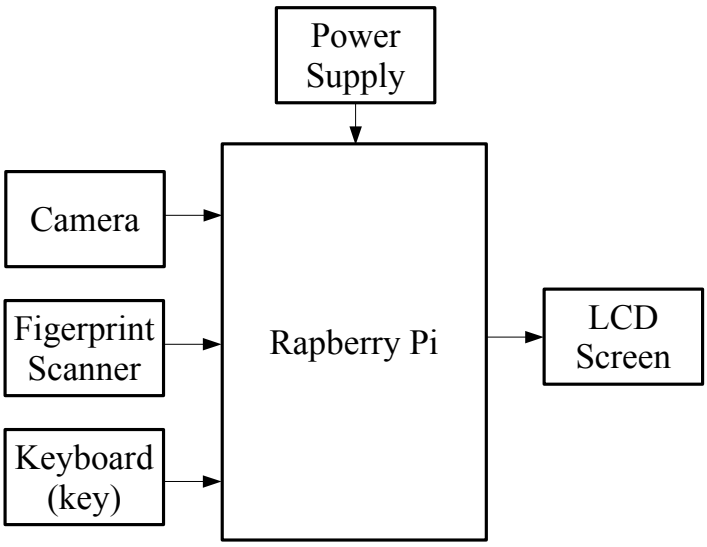


Figure 6.1: Block diagram of the biometric video watermarking system.

The embedding of biometric data into an image has been proposed by many researchers [**Selvy2013**; **Kannan2013**]. The fingerprint biometric features can be easily acquired using a low-cost fingerprint scanner connected to a suitable hardware like the Raspberry Pi [**Sapes2016**].

6.3 The proposed system

The watermarking system presented here is depicted in Figure 6.1. The RP Linux OS running the OpenCV library and Python (Appendix A) makes the implementation straightforward. The hardware elements used in this system are as the following:

- **Raspberry Pi board:** Due to its low cost and low energy consumption, the Raspberry Pi 3 Model B (Appendix B) is used as the central element of the video watermarking system. This board has all the capabilities to implement the video watermarking algorithm: a 64-bit quad-core ARM CPU, which runs at 1.2GHz, 1 GB, 900MHz of RAM, a 400MHz VideoCore GPU, and a multiple I/O ports (USB, GPIO, HDMI, Camera Serial Interface (CSI), Display Serial Interface (DSI)). Along with that, the RP3 is supplied with a modern and powerful Linux operating system (Raspbian) which can facilitate the implementation by providing the necessary drivers and software tools.

- **Pi Camera module:** The video sequences are captured using the Pi camera rev 1.3, which plugs directly on the Raspberry Pi CSI connector and is capable of capturing still images as well as high definition videos. Although it is possible to select high resolution of frames, it is much more practical to select lower resolutions in order to speed up the watermarking process.

- **Fingerprint scanner**: The used module is 5V TTL optical scanner [**Sapes2016**]. It does all of the heavy calculation behind reading, image rendering, feature-finding and searching using an on-board optical sensor and 32-bit DSP. To communicate with the module, a simple serial protocol (UART - 9600 baud) is used. In our algorithm,we use the scanner to retrieve the fingerprint image (and not the template) to use it as the watermark image.

- **Keyboard**: It is a simple matrix Keypad used to enter the user secret key.

- **LCD Screen**: This is a touch screen module designed especially for Raspberry Pi. Its role is to provide a mean of interaction with the RP using the developed GUI.

6.4 The implemented video watermarking method

The proposed video watermarking method is similar to the previous DCT-only method given in Section 4.5 except for two differences:

- The zig-zag scan operation is replaced by a simple column-wise concatenation of each frame in order to transform the channel Y (resulted from RGB to YUV conversion) into one vector. This modification is needed in order to enhance the speed of the video watermarking process on RP. This will affect slightly the transparency of the watermarked video segment, because of the low correlation between the sub-vectors $X_1$and X_2 (Section 3.2) in this case, but this drawback can be compensated by embedding fewer number of bits in each frame. Likewise, the process of watermark extracting is also similar to the one of the DCT-only method given in Figure 5.4, but with using the column-wise concatenation instead of the zig-zag operation.

- The watermark which is a fingerprint image has to be binarized (Figure 6.2) before embedding into the video segment. A hard thresholding binarization with a threshold of 127 is used in the experiments. After that, the general embedding process as in Figure 5.1 is performed using the new binary watermark.

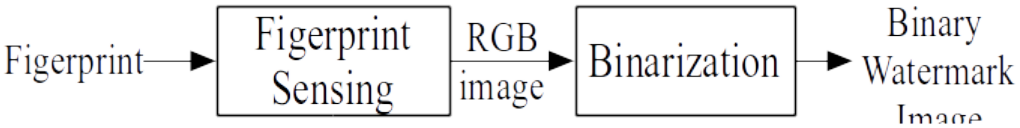


Figure 6.2: Preprocessing of the fingerprint image.

Figure 6.3: From left, the original fingerprint image, the binarized version and the scrambled version.

We should note that this embedding scheme is different from many other biometric methods where the biometric template is used as the user key and not as a watermark itself [**Kannan2013**].

6.5 The video watermarking GUI

The proposed graphical user interface (GUI) for performing video watermarking using the Raspberry Pi Platform is given by the next figure :

The major functions of this GUI are the following:

1. **Start Recording** : instruct the RP to initialize the camera to start the recording of a 300 frames video sequence. Upon finishing, the RP will save the video segment in its file system (on the SD card).

2. **Read Sensor :** the RP will read the fingerprint template from the sensor and save it as an ASCII text file.

3. **Embedding** : the RP reads the template file and converts its characters to a binary sequence in order to fit the specification of embedding algorithm (section 4.5). After that, the system embed this binary sequence into the already-saved video sequence using the user secret key read from the keyboard. The output of this process is the watermarked video segment and it is saved also in the RP system file. The PSNR value is calculated and printed by the GUI in order to asses the quality of the watermarked video sequence.

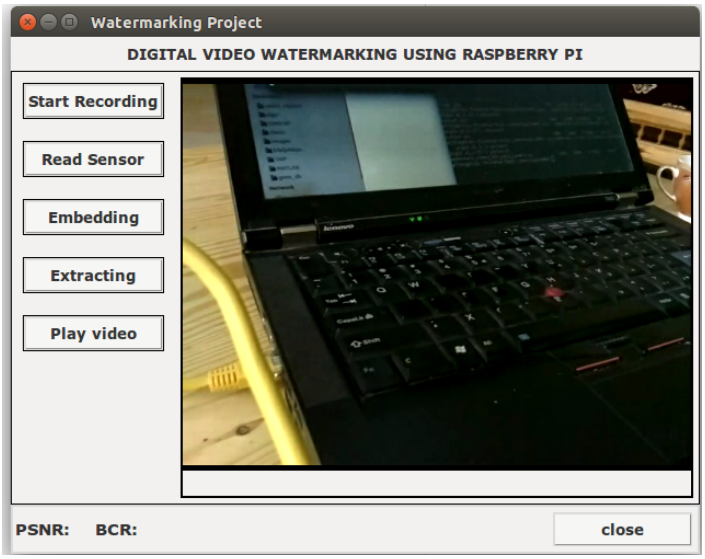


Figure 6.4: The video watermarking GUI.

4. **Extracting**: the user have to re-inter his secret key and re-scan his fingerprint in order to perform this task. Using the secrete key, the system extracts the binary sequence from the embedded video segment and compares it to the one given by fingerprint sensor by calculating the BCR measure. If the BCR value is greater then a pre-defined threshold, the user is considered as authenticated.

6.6 Experiments

In these experiments, video segments of 300 frames are captured (by the camera module) with 288x352 resolution at 30 frames per second in order to be compatible with those used in chapter 4. To evaluate the robustness of the implemented DCT-only video watermarking algorithm, we concentrate only on following video compression attacks : Motion JPEG (MJPEG), MPEG-4, and H.264 standards because of the availability of their libraries on the RP operating system (Appendix B). Notice that the PSNR of the watermarked video are preserved through the embedding factor α to be slightly more than 35 dB.

6.6.1 Time execution

The processing time per function for the watermark embedding and the watermark extracting are given by Tables 6.1 and 5.2 respectively. The main time-consuming functions are given by the the Python Profilers module for the function used in the implementation (Appendix C).

Function	frm_embed	vid_embed	dct	idct
Total time (s)	6.05	27.32	6.03	5.75
N°of calls	200	1	400	400
Time per call (s)	0.03	27.32	0.01	0.01

Table 6.1: Time execution for watermark embedding.

Function	frm_extract	dct	vid_extract
Total time (s)	6.34	4.02	11.73
N°of calls	200	400	1
Time per call (s)	0.03	0.01	11.73

Table 6.2: Time execution for watermark extracting.

First we notice that, the watermark embedding at the frame level (embedding one column of the watermark image) takes only 0.03 s which is acceptable. The same remark can be made regarding the watermark extracting at the frame level. Also, the most time-consuming functions at the watermark embedding and extracting phases are the DCT and the inverse DCT ones which is consistent with the features of transform domain watermarking techniques.

Note that the total watermark embedding and extracting times (27.32s and 11.73s respectively) can be reduced significantly by using the C/C++ language instead of Python but at the expense of the implementation complexity.

6.6.2 Robustness against video compression

Tables 6.3 and 6.4 shows respectively the BCR of the extracted watermark for MJPEG and MPEG4 compression standards of three video segments and for different quality scale. Table 6.5 shows the performance of the method against the more recent and size-efficient type of video compression, i.e, the H.264 standard.

These results confirm the observations made in Chapter 4 about the robustness of the proposed method against video compression even with the usage of column wise concatenation instead of the zigzag operation. The robustness against Motion JPEG and MPEG-4 standards is very good for the acceptable range of the quality factor (Qscal) used in the implementation. For

Quality (Qscal)	1	2	3	4	5
Segment 1	100	100	100	100	99
Segment 2	100	100	100	100	99
Segment 3	100	100	100	100	99

Table 6.3: Robustness against Motion JPEG compression attack.

Quality (Qscal)	1	2	3	4	5
Segment 1	100	100	100	99	97
Segment 2	100	100	100	99	98
Segment 3	100	100	100	98	96

Table 6.4: Robustness against MPEG4 compression attack.

CRF	2	5	10	15	20
Segment 1	100	100	100	99	87
Segment 2	100	100	100	99	84
Segment 3	100	100	100	98	83

Table 6.5: Robustness against H.264 compression attacks.

the H264 standard, the behavior of the implemented method is also similar to the simulation results given in chapter 4; it has very good performance under low compression ratios but it gets limited under high to severe ones.

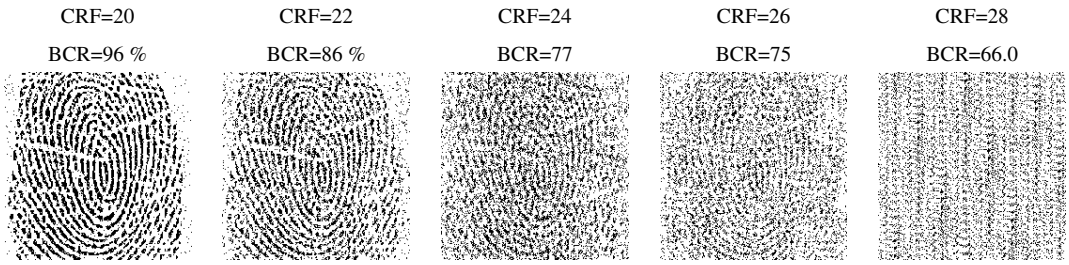


Figure 6.5: The extracted watermark under different types of attacks.

6.7 Conclusion

In this chapter we proposed and implemented a biometric watermarking method based on DCT-based video watermarking method discussed in chapter 4. The hardware and software characteristics of the Raspberry Pi platform make the implementation simple and straightforward. The video segments are captured using the Pi camera module and the fingerprint image is acquired from a fingerprint scanner connected to the RP. A column-wise concatenation is used in the implementation instead of the zig-zag operation in order to accelerate the embedding and extracting process. The whole method is implemented using Python and the OpenCV library. The experimental results validated those found in chapter 4 and proved the feasibility of the implementation on the RP platform.

Chapter 7

GENERAL CONCLUSION AND FUTURE WORK

In this thesis, we have proposed a robust, yet simple watermarking scheme based on the combination of DWT and DCT domains. In the embedding process, a differential technique is performed on two transformed sub-vectors so that the extraction of the watermark is achieved using only the difference of the corresponding watermarked sub-vectors.

Overall, the experimental results demonstrate that our scheme provides excellent robustness against multiple image attacks such as bit-plan removal, cropping, JPEG compression, histogram equalization, low-pass filtering, and noise adding attacks. Besides, the quality of the watermarked image is satisfactory in term of imperceptibility as the PSNR per watermarked image is over 42 dB.

We have also investigated the utility of the combination of the DWT and DCT transforms through the proposition of a relaxed version of our method based only on the DCT transform. In comparison, the DWT-DCT method is more robust than the DCT-only method for a set of attacks such as JPEG compression and low-pass filtering. The results of experiments have showed also that the proposed (DWT-DCT) method has stronger robustness in comparison with two existing watermarking schemes.

We have also extended the proposed approach to video watermarking domain. In order to maintain an acceptable speed for the video embedding and extracting processes, we have simulated only the DCT-based method. In spite of that, the DCT-based video watermarking method has shown a good robustness against a number of attacks such noise adding, filtering and video compression.

Finally, we have implemented the DCT-based video watermarking using the Raspberry Pi. We used a fingerprint image as a user watermark in order to increase the security of watermarking system. The results of this implementation exhibit good performances in terms of transperency and robutness especially against video compression which confirms the results of the simulations.

The research presented in this thesis indicates many direction to pursue further research in this domain. We summarize some of them as follows:

- An automatic (and perhaps adaptive) technique for the selection of the gain factor value, need to be developed to have better control on both imperceptibility and robustness of the scheme.

- Decompostion by other types of wavelets instead of the wavelet of Haar can be made. We suggest to use wavelet packets or the discrete stationary wavelet transform in order to increase the capacity of the method.

- Investigation of the best color space for embedding the watermark in color image and video watermarking is also possible.

- Research on compression domain video watermarking for th H.264 standard is also feasible. Using a similar approach presented in this thesis, a robust technique can be proposed in order to enhance the robustness against H.264 compression.

- For the video watermarking implementation, we suggest to use an FPGA hardware in order to gain a real-time performance watermark embdding and extracting.

Appendix A

Software tools used in the development of the project

Most of the tools used in the development of this doctoral project are free software: we used Octave in the development of the image watermarking techniques in Chapter 4 and Python in the implementation of the video watermarking method in Chapter 6. Matlab was used only in Chapter 5 to simulate the video watermarking method, because of the lack of video processing toolbox in Octave at present. Here's a brief description of them:

Octave

Octave is a high-level language, primarily intended for numerical computations which has the following characteristics www.gnu.org/software/octave:

- It provides a powerful mathematics-oriented syntax with built-in plotting and visualization tools,
- Free software, runs on GNU/Linux, macOS, BSD, and Windows,
- Drop-in compatible with many Matlab scripts,

Python

Python is a widely used high-level programming language for general-purpose programming. An interpreted language, Python has a design philosophy which emphasizes code readability, and a syntax which allows programmers to express concepts in fewer lines of code than possible in languages such as C++ or Java www.python.org.

OpenCV

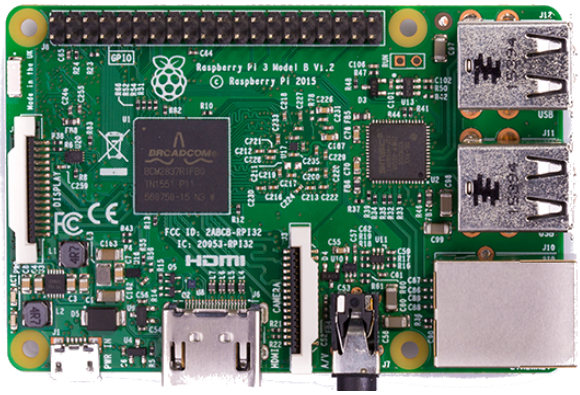
OpenCV (Open Source Computer Vision Library) is an open source computer vision library initiated by Intel to incorporate image processing into a wide variety of coding languages. It has C++, C, and Python interfaces running on Windows, Linux, Android and macOS. The OpenCV library contains over 500 functions that span many areas in vision, including medical imaging, security, user interface, camera calibration, and machine learning www.opencv.org.

Appendix B

The Raspberry Pi platform

Raspberry Pi, as shown in Figure 5.2, is a single-board computer having a size as small as a credit card from the Raspberry Pi foundation www.raspberrypi.org. In 2006, A group of academics and engineers from the university of Cambridge, decided to develop a very small computer which everyone could afford to buy to create learning environment in programming. The Raspberry Pi project became promising with the appearance of cheap and powerful mobile processors (ARM architecture) with many advanced features allowing a possible development of Raspberry Pi which was continued under specially created Raspberry Pi foundation with the first product launched in 2012.

The Raspberry Pi 3 (used in the implementation) is the third generation Raspberry Pi. It replaced the Raspberry Pi 2 Model B in February 2016. Its main specifications are the following:



The Raspberry Pi 3

- 1.2GHz 64-bit quad-core ARMv8 CPU

- 1GB RAM
- VideoCore 4 (300 MHz, Full HD 1080p)
- 40 GPIO pins
- 802.11n Wireless LAN
- Ethernet port
- Bluetooth 4.1
- 4 USB ports
- Full HDMI port

In addition to that, the RP3 platform has low power consumption (260 mA, 1.4W at Idle), a Camera interface (CSI), a Display interface (DSI), etc. All these specification, make it an ideal choice for our video watermarking implentation.

Raspberry Pi runs on special derivatives of Linux OS. Raspbian is the OS which is specially and most developed for Raspberry Pi. The OS is based on Debian Wheezy armhf (www.debian.org) which uses floating point registers available in hardware for floating point parameters operation. Raspbian contains Debian packages along with additional packages built for Raspberry Pi, for example, camera module software, Wolfram Mathematica and many others.

Appendix C

The Python code of the video watermarking method

The Python code of the DCT-based video watermarking method presented in Chapter 6 is given by the following listing (Without the GUI widgets):

```
from cv2 import *
import os
import sys
import cProfile
from numpy import *
import math
##### Global Variables definition #####

alpha=0.4          # Embedding coeff
NF=300;            # Nbre of frames
Fw=352;            # Frame width
Fh=288;            # Frame height
a=10;b=Fw*Fh/2;
key1=random.permutation(range(a,b)) # random points in which the watermark bits are inserted
key2=random.permutation(range(0,NF)) # random frames to be watermarked
key2=key2[0:200]   # 200 from NF frames to be embedded (watermark image is 200X200)
key3=25                    # Number of itr of Arnold trasform
##### convert bit's 0 of th watermark to -1 #####
def zeros2ons(col):
        col= float32(col)
        for k in range(col.size):
                if col[k]==0:
                        col[k]=-1
        return col

##### Arnold cat-map #####
```

```
def arnold(img, itr ):
        assert img.shape[0] == img.shape[1], "Input␣image␣must␣be␣square"
        N=img.shape[0]
        x,y = meshgrid(range(N),range(N))
        for i in range( itr ):
                xmap = (2*x+y) % N
                ymap = (x+y) % N
                x=xmap
                y=ymap
        return (img[xmap,ymap])
##### Inverse Arnold cat-map #####
def iarnold(img, itr ):
        assert (img.shape[0] == img.shape[1]), "Input␣image␣must␣be␣square"
        N=img.shape[0]
        x,y = meshgrid(range(N),range(N))
        for i in range( itr ):
                xmap = (2*x-y) % N
                ymap = (-x+y) % N
                x=xmap
                y=ymap
        return (img[xmap,ymap])
##### converting negative values of
##### the extracted watermark to -1 #####
def neg2zero(col ):
        col= float32 (col)
        for k in range(col.size ):
                if col[k]<0:
                        col[k]=0
                else :
                        col[k]=1
        return col
##### Bit Correct Error function #####
def BCR(A,B):
        error =0
        for a, b in zip(A, B):
                if a == b:
                        error += 1
        return 100*error/len(A)
##### PSNR value of two gray images #####
def PSNR(imageA, imageB):
        err = sum((imageA.astype("float32") - imageB.astype("float32")) ** 2)
```

```
68          err /= float(imageA.shape[0] * imageA.shape[1])
69          max_frm=imageA.max()
70          psnrfin =10*(math.log10((max_frm**2)/err))
71          return psnrfin
72  ##### PSNR value of two color frames #####
73  def PSNRCOLOR(fr1, fr2):
74          p=(PSNR(fr1 [:,:,0], fr2 [:,:,0])+ PSNR(fr1 [:,:,1], fr2 [:,:,1])+ PSNR(fr1 [:,:,2], fr2 [:,:,2]))/3.0
75          return p
76  ##### PSNR value of two video segments #####
77  def vid_psnr( file1 , file2 ):
78          cap1 = VideoCapture( file1 )
79          cap2 = VideoCapture( file2 )
80          k=0;P=0;
81          while(cap1.isOpened() and k<NF):
82                  ret , frame1 = cap1.read()
83                  ret , frame2 = cap2.read()
84                  P+=PSNRCOLOR(frame1,frame2)
85                  k+=1
86          cap1. release ()
87          cap2. release ()
88          return P/NF
89  ##### Video compression attacks
90  ##### using avconv (ffmpeg based) Linux command #####
91  def vid_attack ():
92          os.system("cp␣output.avi␣output1.avi")
93          #os.system("avconv -i output1.avi -c:v mjpeg -q:v 2 -an output.avi -y")  # MJPEG compression
94          #os.system("avconv -i output1.avi  -q:v 3  -c:v mpeg2video output.avi -y")# MPEG2 compression
95          os.system("avconv␣-i␣output1.avi␣-q:v␣3␣-c:v␣libxvid␣output.avi␣-y")  # MPEG4 compression
96          #os.system("avconv -i output1.avi -c:v libx264 -crf 2  output.avi -y")  # h.264 compression
97  ##### Read the image watermark
98  ##### and convert it to the binary #####
99  def Read_Sensor():
100         global W
101         W=imread("33.jpg",IMREAD_GRAYSCALE)
102         ret ,W = threshold(W,127,255,THRESH_BINARY)
103         imwrite(' original .png',W)
104         W=arnold(W,key3)
105         W=W/255;
106         return W
107 ##### Embedding one frame with one column
108 ##### of the binary wtermark image #####
```

```
109 def frm_embed(frame,col,key1,alph ):
110     frameYUV = cvtColor(frame, COLOR_BGR2YUV) # convert to YUV
111     gray_ch=frameYUV[:,:,0]
112     col=zeros2ons(col );
113     (L,H)=shape(gray_ch)
114     k=i=j=0
115     v=gray_ch.reshape(L*H) # convert 2d to 1d vector
116     v=v.astype(" float64 ")
117     v1=v[0::2]
118     v2=v[1::2]              # sub-sampling
119     V1=dct(v1)
120     V2=dct(v2)              # dct
121     i=0
122     ###### embeding process #####
123     while   i<MW :
124             pos=key1[i]
125             pos=int(pos)
126             V1[pos]=0.5*(V1[pos]+V2[pos] )+ (alph*col[i])
127             V2[pos]=0.5*(V1[pos]+V2[pos] )-( alph*col[i])
128             i=i+1
129     v1w=idct(V1)
130     v2w=idct(V2)          # inverse dct
131     vw=zeros((L*H,1))
132     vw[0::2]=v1w
133     vw[1::2]=v2w   # up-sampling
134     gray_ch_W=float32(vw.reshape((L,H))) # convert 1d vectorto 2d image
135     frameYUV[:,:,0]=gray_ch_W # reconstruct the YUV watermarked frame
136     frameRGB = cvtColor(frameYUV, COLOR_YUV2BGR) # reconstruct the RGB watermarked frame
137     return frameRGB
138 #####   End of frame embedding #####
139 #####   Extracting one column of watrmark
140 #### image from one frame #####
141 def frm_extract (frame,key1):
142     frameYUV = cvtColor(frame, COLOR_BGR2YUV)
143     gray_ch=frameYUV[:,:,0]
144     ext_M=zeros(MW)
145     (L,H)=shape(gray_ch)
146     x=zeros(L*H/2)
147     y=zeros(H*L/2)
148     v=gray_ch.reshape(L*H)
149     x=v[0::2]
```

```
150     y=v[1::2]
151     X=dct(x)
152     Y=dct(y)
153     i=0
154     while i< MW:
155         pos=int(key1[i])
156         ext_M[i]=(X[pos] -Y[pos])
157         i=i+1
158     ext_M=neg2zero(ext_M)
159     return ext_M
160 ##### End of frame extracting #####
161 ##### Recording a raw video segment using RP
162 #### NF frames, size : FhXFw, fps:30 #####
163 def startrecording ():
164         cap = VideoCapture(0)
165         cap.set(3,Fw)
166         cap.set(4,Fh)
167         out = VideoWriter('myvideo.avi',0,30.,(Fw,Fh))
168         k=0
169         while(cap.isOpened() and k<NF):
170                 ret, frame = cap.read()
171                 k+=1
172                 #print k
173                 out.write(frame)
174         else:
175                 cap.release()
176                 out.release()
177 ##### End of recording #####
178 ##### Playing the video segment #####
179 def play_video():
180         cap = VideoCapture('output.avi')
181         k=0
182         while (cap.isOpened() and k<NF):
183                 ret, fram = cap.read()
184                 if (ret==False):
185                         break
186                 imshow('video', fram)
187                 waitKey(50)
188                 k=k+1
189 ##### End playing video segment #####
190 ##### Embedding th whole video segment with
```

```
191 ##### the binary image watermark #####
192 def vid_embed():
193   cap = VideoCapture('myvideo.avi')
194   Nfrm = int(cap.get(CAP_PROP_FRAME_COUNT))
195   Lfrm = int(cap.get(CAP_PROP_FRAME_WIDTH))
196   Hfrm = int(cap.get(CAP_PROP_FRAME_HEIGHT))
197   fps =cap.get(CAP_PROP_FPS)
198   maax=int((Lfrm*Hfrm/10))
199   out = VideoWriter('output.avi',0, fps ,(Lfrm,Hfrm)) # The watermarked segment
200   k=k1=0;
201   while(cap.isOpened() and k<NF):
202             ret , frame = cap.read()
203             if (( ret==True) and (k in key2)):
204                       col=W[:,k1]
205                       frame1=(frame.astype("float32"))/255.0
206                       frm=frm_embed(frame1,col,key1,alpha)
207                       out.write(uint8(255*frm))
208                       k=k+1
209                       k1=k1+1
210             elif (( ret==True) and (k not in key2)) :
211                       out.write(frame)
212                       k=k+1
213             else:
214                       break
215   cap.release()
216   out.release()
217 ##### End of video segment embedding #####
218 ##### Extracting thw whole binary image
219 ##### watermark from the watermarked segment #####
220 def vid_extract():
221   cap1 = VideoCapture('output.avi')
222   Nfrm11 = cap1.get(CAP_PROP_FRAME_COUNT)
223   ext_WM=zeros((200,200))
224   k=k1=0
225   while(cap1.isOpened() and k<NF):
226       ret1 , frame1 = cap1.read()
227       if (( ret1==True) and (k in key2)):
228                       frame1=frame1.astype("float32")/255.0
229                       extractd_M1=(frm_extract(frame1,key1))
230                       ext_WM[:,k1]=(extractd_M1)
231                       k=k+1
```

```
232                     k1=k1+1
233         elif  (ret1==True):
234                     k=k+1
235         else :
236                     break
237     print  "The␣BCR=", BCR(W.ravel(),ext_WM.ravel()) # calculate the BCR value
238     ext_WM=iarnold(ext_WM,key3)                     # descramble the watermark image
239     imwrite('extratedmark.png', uint8(255*ext_WM))  # save the watermark image
240     print  "The␣PSNR=", round(vid_psnr('myvideo.avi','output.avi'),1)  # calculate the PSNR value
241     cap1.release()
242 #####   End of watermark extracting  #####
```