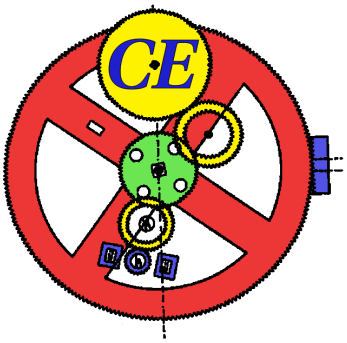

Computer Engineering
Mekelweg 4,
2628 CD Delft
The Netherlands
http://ce.et.tudelft.nl/

2009

MSc THESIS

Digital Image Watermarking Robustness: A Comparative Study

Mitra Abbasfard

CE-MS-2009-15

Abstract

Watermarking is the process of inserting predefined patterns into multimedia data in a way that the degradation of quality is minimized and remain at an imperceptible level. Many digital watermarking algorithms have been proposed in special and transform domains. The techniques in the spatial domain still have relatively low-bit capacity and are not resistant enough to lossy image compression and other image processing operations. For instance, a simple noise in the image may eliminate the watermark. On the other hand, frequency domain-based techniques can embed more bits for watermark and are more robust to attack. Some transforms such as Discrete Cosine Transform (DCT) and Discrete Wavelet Transform (DWT) are used for watermarking in the frequency domain. In this thesis, we compare the DCT watermarking algorithms and especially the DWT watermarking algorithms based on robustness criteria. In other words, the robustness of different transform watermark algorithms is evaluated by applying different attacks. One main reason to consider the DWT watermarking algorithms is that several multimedia standards such as the JPEG2000 and MPEG-4 are based on the DWT. These new standards brought new requirements such as progressive, low bit rate transmission, and region-of-interest coding.

TU Delft
Delft University of Technology

Faculty of Electrical Engineering, Mathematics and Computer Science

Digital Image Watermarking Robustness: A Comparative Study

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

by

Mitra Abbasfard
born in Shiraz, Iran

Computer Engineeringtrre
Department of Electrical Engineeringrere
Faculty of Electrical Engineering, Mathematics and Computer Scienceerrer
Delft University of Technologyrertre

Digital Image Watermarking Robustness: A Comparative Study

by Mitra Abbasfard

Abstract

Watermarking is the process of inserting predefined patterns into multimedia data in a way that the degradation of quality is minimized and remain at an imperceptible level. Many digital watermarking algorithms have been proposed in special and transform domains. The techniques in the spatial domain still have relatively low-bit capacity and are not resistant enough to lossy image compression and other image processing operations. For instance, a simple noise in the image may eliminate the watermark. On the other hand, frequency domain-based techniques can embed more bits for watermark and are more robust to attack. Some transforms such as Discrete Cosine Transform (DCT) and Discrete Wavelet Transform (DWT) are used for watermarking in the frequency domain. In this thesis, we compare the DCT watermarking algorithms and especially the DWT watermarking algorithms based on robustness criteria. In other words, the robustness of different transform watermark algorithms is evaluated by applying different attacks. One main reason to consider the DWT watermarking algorithms is that several multimedia standards such as the JPEG2000 and MPEG-4 are based on the DWT. These new standards brought new requirements such as progressive, low bit rate transmission, and region-of-interest coding.

Laboratory : Computer Engineering
Codenumber : CE-MS-2009-15

Committee Members :

Advisor: Dr. K.L.M. Bertels , CE, TU Delft

Member: Dr. K.L.M. Bertels , CE, TU Delft

Member: Dr. J.S.S.M. Wong, CE, TU Delft

Member: Dr. Reza Hassanpour, Cankaya University

In memory of my father Mohammad Esmaeil Abbasfard and to my dear mother Mehr Mah Nekooie Mehr

Contents

List of Figures

List of Tables

Acknowledgments

First, my special thanks goes to Prof. Stamatis Vassiliadis who encouraged and helped me to continue my study in computer engineering. I always remember all his guidance, support, kindness, and friendliness. God bless him.

I am especially grateful for support, patience, time, and help of my advisor Dr. Koen Bertels during my study. I would like to thank Dr. Stephon Wong, and Dr. Reza Hassanpour for the time and help they also gave. I would also like to thank all CE members for their collaboration.

My special thanks goes to all my Iranian friends warmly and also my friend Helen for their support in all aspect during my residence in Delft. Finally but certainly not least, I am grateful to my family especial my mother for her unlimited and unconditional love, patience and support in my whole life and to my husband Asad for all support, encouragement, and patience during my study. My deepest thanks and love to my little angel Yasamin for her love, patience, and her pretty smile which always give me hope, confidence, and energy in my life.

Delft, *Mitra Abbasfard*
June 2009

1 Introduction

Development of compression algorithms for multimedia data such as MPEG-2/4 and JPEG standards, and increase in the network data transmission speed have allowed widespread use of applications, which rely on digital data. In other words, digital multimedia data are rapidly spreading everywhere. On the other hand, this situation has brought about the possibility of duplicating and/or manipulating the data. To keep on with the transmission of data over the Internet the reliability and originality of the transmitted data should be verifiable. It is necessary that multimedia data should be protected and secured.

One way to address this problem involves embedding an invisible data into the original data to mark ownership of them. There are many techniques for information hiding, which can be divided into different categories such as convert channels, steganography, anonymity, and watermarking [23]. Convert channels techniques were defined in the context of multilevel secure systems. Convert channels usually handle properties of the communication channels in an unexpected and unforeseen way in order to transfer data through the medium without detection by anyone other than the entities operating the covert channel. Steganography is about preventing the detection of an encrypted data, which has been protected by cryptography algorithms. Anonymity is a technique to find ways to hide the meta content of tranmitted messages such as sender and the recipients. Digital watermarking has an extra requirement of robustness compared to steganography algorithms against possible attacks. It should be also noted that watermarking is not intended for protecting of the content of a message, and hence it is different from cryptography. In this thesis we focus on the robustness of the digital watermarking algorithms in the transform domain against common attacks.

This chapter is organized as follows. Section 1.1 presents the motivation for this work. Our project goals are identified in Section 1.2. Section 1.3 concludes the introduction chapter with an overview of thesis organization.

1.1 Motivation

There are different algorithms in the spatial and transform domains for digital watermarking. The techniques in the spatial domain still have relatively low-bit capacity and are not resistant enough to lossy image compression and other image processing. For instance, a simple noise in the image may eliminate the watermark data. On the other hand, frequency domain-based techniques can embed more bits for watermark and are more robust to attack. Some transforms such as Discrete Cosine Transform (DCT) and Discrete Wavelet Transform (DWT) are used for watermarking in the frequency domain.

Most DCT-based techniques work with 8×8 blocks. These transforms are being used in several multimedia standards such as MEPG-2, MPEG-4, and JPEG2000. In addition, different watermark algorithms have been proposed using DCT and DWT. In considering the attacks on watermarks, the robustness feature of an algorithm becomes very important. In this regard, we classify a watermark method as robust if the watermark data embedded by that algorithm in an image or any other data, cannot be damaged or removed without destroying or damaging the data itself. Therefore, an attack is successful if it can eliminate the watermark without damaging the image itself. The question is, which transform watermark algorithms are more robustness to different attacks compared to other techniques? We perform a comparative study on different transform watermark algorithms and compare their robustness.

In fact the robustness of the algorithms is dependent on the frequency at which the watermark data is added. We have performed evaluation by having in mind that the embedded watermark should be invisible so we have kept the Peak Signal to Noise Ratio (PSNR) value of the images constant at 35dB and compared the robustness of the different methods.

1.2 Project Goals

Not all watermarking methods are suitable for a particular application. The suitability of a method depends on the type of data, the processing or transformations applied on data, the lossy or lossless type of compressions used, the key type used, etc. An objective and quantitative measurement on the suitability of each method for a given application is of great importance and usefulness. The project work will include the following steps:

- We study the main DCT and DWT watermark algorithms. The transform watermark algorithms are more robust than spatial domain.
- We study different watermark attacks such as removal attacks, and geometric attacks.
- We implement several important transform watermark algorithms and we test their robustness using different attacks.

1.3 Thesis Organization

The thesis is organized as follows. In Chapter 2, we discuss the basic information about digital watermarking and different requirements that a digital watermarking algorithm should consider. In addition, several applications of digital watermarking, a classification of different watermarking algorithms, and some hardware solutions are presented. In Chapter 3, we discuss transform domain watermarking, specially focusing on DCT and DWT in more detail. We explain the concept of DWT and different ways to implement it. Different watermarking algorithms based on DWT are discussed in Chapter 4. In Chapter 5 some important waterwarking algorithms based on the DCT and DWT are

evaluated and their robustness are compared with each other. We present the summary and the future work in Chapter 6.

2 Digital Watermarking

The introduction of watermarking and the related publications date back to 1979. However, it was only in 1990 that it gained large international interest. Digital watermarking involves embedding watermark data into original information. In other words, a watermark is a pattern of bits inserted into multimedia data such as digital image, audio or video file that helps to identify the file's copyright information (author, rights, etc.). A simple example of a digital watermark may be a visible signature or seal placed over an image to determine the owner of that image. The name "watermark" is derived from the faintly visible marks imprinted on organizational stationery. Unlike printed watermarks, which are intended to be somewhat visible, digital watermarks are designed to be completely invisible, or in the case of audio clips, inaudible. In addition, the bits representing the watermark must be scattered throughout the file in such a way that they cannot be identified and manipulated. The embedding technique must keep the original information perceptually unchanged and the watermark data should be detected by an extraction algorithm.

This chapter in organized as follows. In Section 2.1, we discuss the basic information about the different requirements that a digital watermark algorithm should have. In Section 2.2 several applications of digital watermarking are discussed followed by Section 2.3 where we present a classification of different watermarking algorithms. We explain single and multiple watermarking algorithms in Section 2.4 and we also describe several commom attacks on digital watermarking in Section 2.5. At the end of this chapter, we describe several hardware solutions to implement digital watermarking algorithms in Section 2.6.

2.1 The Requirements of Digital Watermarking

It is clear that watermark information cannot be stored in the file header because anyone with a computer or a digital editing workstation would be able to convert the information to another format and remove the watermark at the same time. Thus the watermark should really be embedded to the multimedia signals.

Figure 2.1 depicts a watermark embedding scheme. The input data consists of the original multimedia and watermark data. The watermark data is produced by a production algorithm, which may uses a secret key, a signature, or a combination of several secret keys and the original data. In other words, the standard procedure of watermarking starts with an embedding stage. At this stage a cover media file is considered and the watermarking information is embedded using a secret key. Both watermark information and secret key have to be selected and available prior to the embedding stage[28, 9]. At

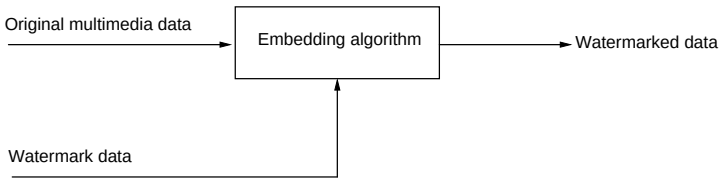

Figure 2.1: Block diagram of a watermark embedding scheme.

last, the output of the watermark embedding process is the watermarked data.

The most important requirements for digital watermarking are summarized below. However, the relative importance of these properties depend on the application.

- **Transparency (invisibility):** This refers to perceptual similarity between the watermarked image and the original image. The watermark should be imperceptible. It means that no visual or audio effect should be perceived by the end user. The watermark should not degrade the quality of the content, but in some applications we may accept a little degradation to have higher robustness or lower cost. Sometimes watermark is embedded to data in the way that can be seen without extraction. We called this visible watermark. The example of visible watermark are logos.

- **Robustness:** A watermark algorithm is called robust if it can survive after common signal processing operations. In other words, it is detectable after common signal processing operations such as lossy compression, spatial filtering, translation, and rotation operations. It is important to know that watermark can be robust against one process and fragile against another. A watermark just needs to survive the common signal processing that accrues between the time of embedding and detection of the watermark, and it also depends on the application. If the watermark data is placed in significant coefficients of an image then the robustness against image distortion is better achieved. This is because those coefficients do not change so much after common image processing and compression operations [12]. Contrary to robust watermark, a fragile watermark is not designed to be robust.

- **Capacity:** A watermarking system must allow for a useful amount of information to be embedded into an image. The amount of information that can be embedded in a watermarked image is called data payload. The data payload in image watermarking means the number of bits encoded with the image. The payload of the embedded watermark information must be sufficient to enable the envisioned application.

- **Security:** The watermark must withstand attacks that are aimed directly at the embedded information. It must not be possible for an attacker to delete the watermark without rendering the multimedia data unusable. Especially it must not be possible to retrieve or even modify the watermark without knowledge of the secret watermark key.

- **Complexity:** Depending on the application, the insertion is done only once and can be performed off-line. Consequently, the complexity of encoding plays a less important role than the complexity of the decoding. For example, we may have to need real-time decoding. For this reason, the complexity of the watermark algorithms should be in a simple form.

2.2 Application Area

Digital watermarking has so many applications. In this section, we discuss some more common application of digital watermarking.

Copyright Protection: One of the most common applications of digital watermarking is copyright protection specially in image watermarking and it is to insert copyright information (the rules and data of using and copying) into digital object without loss of quality. This kind of applications needs high robustness.

Owner Identification/Proof of Ownership: The identification information of the content owner can be embedded as a watermark data into the original data to prove the ownership. This application requires high level of security.

Copy Control: One of the techniques of copy prevention is to have a copy and consumer control mechanism to prevent illegal copying or recording of the content by inserting a never-copy watermark or limiting the number of times of copying.

Authentication /Content Verification: In authentication the goal is to be able to detect any change or modification of the data so that the information required to authenticate the content should be watermarked. This can be possible through the fragile watermark, which has low robustness to any modification.

Transactional Watermarks/Fingerprinting: The main challenge in fingerprinting is to trace the source of illegal copies so that the owner can embed of a different watermark key into each copy that distributed to a different customer.

Broadcast Monitoring: Advertisers use this kind of application to ensure that the commercials are aired by the broadcasters at the time and location that they want according to the contracts. Watermarks can be embed in any type of data to broadcast on the network by automated systems, which are able to monitor distribution channels to track the content in the time and the place that they appear.

2.3 Classification of Watermark Algorithms

In this section we discuss different classification of watermarking algorithms focusing on the domain in which watermark data is embedded.

2.3.1 Taxonomy of Watermarking

There are different classification of digital watermark algorithms [17, 38]. First, watermark techniques can be divided into four groups according to the type of data to be watermarked.

- Text watermarking
- Image watermarking
- Video watermarking
- Audio watermarking

Second, based on human perception, watermark algorithms are divided into two categories:

- Visible watermarking.
- Invisible watermarking.

Visibility is associated with perception of the human eye so that if the watermark is embedded in the data in the way that can be seen without extraction, we call the watermark visible. Examples of visible watermarks are logos that are used in papers and video. On the other hand, an invisible watermarking cannot be seen by human eye. So it is embedded in the data without affecting the content and can be extracted by the owner or the person who has right for that. For example images distribute over the internet and watermarked invisible for copy protection.

Third, watermark algorithms are classified based on information for detection. They are as follows.

- Blind or public watermarking: In public watermarking, there is no need for original signal during the detection processing to detect the watermark. Only the secret key is required. For example, in image blind watermarking we do not need the original image.
- Non-blind or private watermarking: In non-blind or private watermark, original signal is required for detection the watermark.
- Semi-blind watermarking: In semi-blind watermarking, sometimes we may need some extra information for detecting the watermark. Some watermarking require access to the original signal just after adding the watermarking, which is called published watermarked signal. This form of watermarking is called semi-blind watermarking.

Finally, based on processing-domain, watermark techniques can be divided into:

- Spatial domain: A watermark technique based on the spatial domain, spread watermark data to be embedded in the pixel value. These approaches use minor changes in the pixel value intensity. The simplest example of the former techniques is to embed the watermark in the least significant bits of image pixels [32]. In other words, significant portions of low frequency componenets of images should be modified in order to insert the watermark data in a reliable and robust way. As another example, an image is divided into the same size of blocks and a certain watermark data is added with the sub-blocks [35].
- Transform domain: To have imperceptibility as well as robustness, adding of watermark is done in transform domain. In this method, transform coefficients are modified for embedding the watermark. Transform domain is also called frequency domain because values of frequency can be altered from their original. The most important techniques in transform domain are Discrete cosine transform (DCT) and Discrete Wavelet Transform (DWT).

Additionally, classification can be based on the robustness feature. Different techniques of this category are as follows.

- Robust watermark: One of the properties of the digital watermarking is robustness. We call a watermark algorithm robust if it can survive after common signal processing operations such as filtering and lossy compression.
- Fragile watermark: A fragile watermark should be able to be detected after any change in signal and also possible to identify the signal before modification. This kind of watermark is used more for the verification or authenticity of original content.
- Semi-fragile watermark: Semi-fragile watermark is sensitive to some degree of the change to a watermarked image.

Furthermore, from application point of view, watermark techniques can be grouped as source based or destination based. In source based, all copies of a particular data have a unique watermark, which identifies the owner of that data, while in the destination based, each distributed copy is embedded using a unique watermark data, which identifies a particular destination. Figure 2.2 depicts different classification for digital watermarking algorithms.

In the rest of the section, we discuss processing-domain, spatial and transform in more detail.

2.3.2 Spatial Domain Watermarking

Spatial domain watermark algorithms insert watermark data directly into pixels of an image [20]. For example, some algorithm insert pseudo-random noise to image pixels.

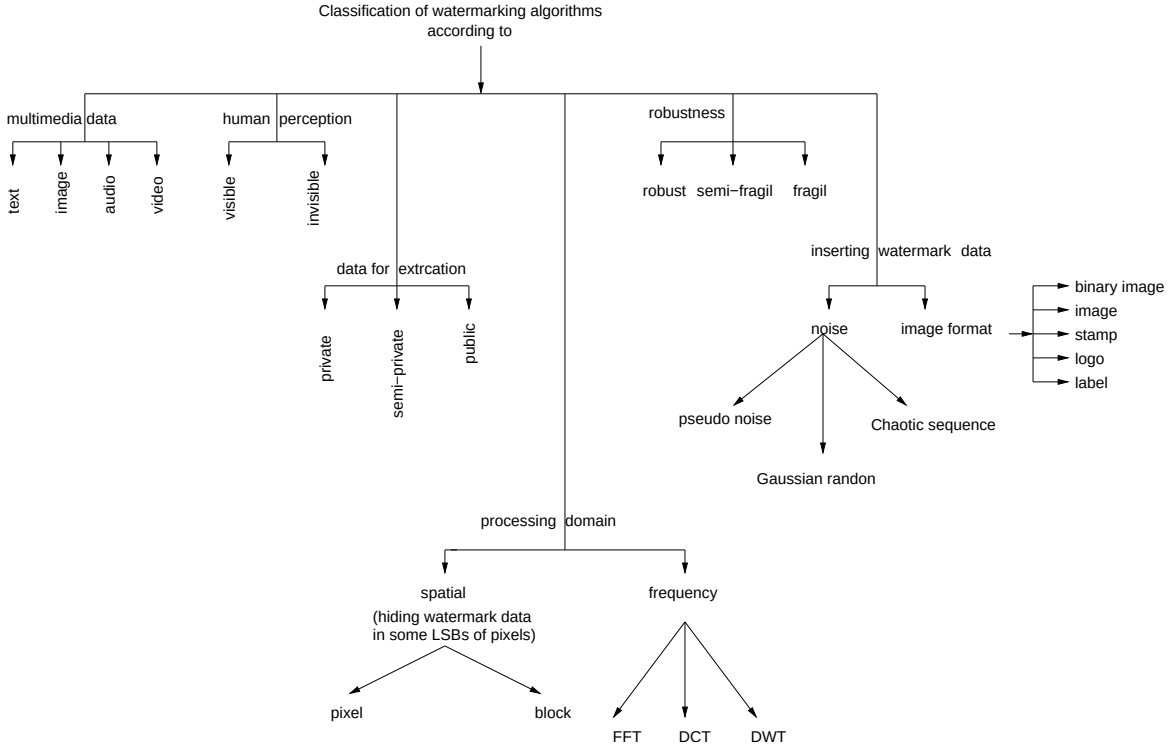

Figure 2.2: Classification of watermarking techniques.

Other techniques modify the Least Significant Bit (LSB) of the image pixels. The invisibility of the watermark data is obtained on the assumption that the LSB bits are visually insignificant. There are two ways of doing an LSB modification. There are some methods to change the LSB bits. The LSB of each pixel can be replaced with the secret message or image pixels may be chosen randomly according to a secret key. Here is an example of modifying the LSBs, suppose we have three R, G, and B component in an image. Their value for a chosen pixel is green $(R, G, B) = (0, 255, 0)$. If a watermark algorithm wants to hide the bit value 1 in R component then the new pixel value has components $(R, G, B) = (1, 255, 0)$. As this modification is so small, the new image is to the human eye indistinguishable from the original one.

Although this spatial domain techniques can be easily used on almost every image, they have the following drawbacks. These techniques are highly sensitive to signal processing operations and can be easily damaged. For example, lossy compression could completely defeat the watermark. In other words, words, watermarking in the spatial domain is easy to destroy using some attacks such as low-pass filtering. As a result, transform domain watermarking algorithms are used.

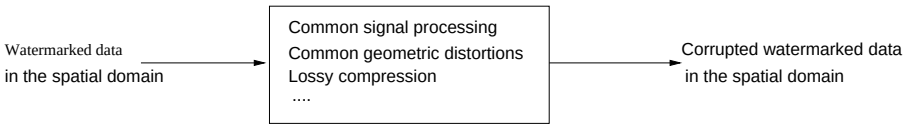

Figure 2.3: Common processing operations on a watermarked data in the spatial domain.

2.3.3 Transform Domain Watermarking

Transform domain watermarking embed watermark data into the transformed image. Transform domain algorithms have many advantages over spatial domain algorithms [12]. For example, Figure 2.3 depicts common processing operations on a watermarked data in the spatial domain. Common signal processing includes operations such as upsampling, downsampling, quantization, and requantization. Rotation, translation, and scaling are common geometric operations. Lossy operation is an operation to remove some unimportant parts of the data. Most of the processing for this category takes place in the transform domain and eliminates high-frequency values. As can be seen in Figure 2.3, those operations corrupts the watermark data, which has been embedded into the original data.

In addition, the techniques in the spatial domain still have relatively low-bit capacity and are not resistant enough to lossy image compression and other image processing [11]. For instance, a simple noise in the image may eliminate the watermark. As another example, a watermark data placed in the high-frequency values can be easily eliminated with little degradation of the image by any low-pass filtering.

On the other hand, transform-domain watermarking techniques are typically much more robust to image manipulation compared to the spatial domain techniques. This is because the transform domain does not use the original image for embedding the watermark data. In addition, a transform domain algorithm spreads the watermark data over all part of the image. Additionally, frequency domain-based techniques can embed more bits for watermark and are more robust to attack. Furthermore, most of the images are avaliable in the transform domain.

Some transforms such as DCT and DWT are used for watermarking in the frequency domain. Most DCT-based techniques work with 8×8 blocks [12]. In this thesis, we compare the DCT watermarking algorithms and especially the DWT watermarking algorithms based on robustness criteria. One main reason to consider the DWT watermarking algorithms is that several multimedia standards such as the JPEG2000 and MPEG-4 are based on the DWT [25, 21]. These new standards brought new requirements such as progressive, low bit rate transmission, and region-of-interest coding.

2.4 Single and Multiple Bit Watermarking Algorithms

Different watermarking algorithms can also be divided into two groups namely, single bit and multiple bits algorithms. In the single bit watermark algorithms, the watermark

pattern consists of the integers randomly selected from $-1, 0, 1$. The watermark pattern is created using a secret key. This secret key is used as an input key to the random number generator. When adding this type of watermark to the image, the care should be taken to uniformly distribute the energy in the pattern. Equation (2.1) shows how the 1-bit watermark is added to the image.

$$I'(x, y) = I(x, y) + kw(x, y). \tag{2.1}$$

where k is the watermark gain factor and a number between 0 and 1 ($0 < k < 1$), $w(x, y)$ is the watermark pattern, $I(x, y)$ is the original image and $I'(x, y)$ is the watermarked image. Single bit watermarking can also be applied in the transform domain to the coefficients of the transformed image. Equation (2.2) indicates this type of watermark embedding.

$$F'(x, y) = F(x, y)(1 + kw(x, y)). \tag{2.2}$$

where $F'(x, y)$ is the watermarked coefficient, and $F(x, y)$ is the original transform coefficients of the image. During the detection in these types of watermarking, either the watermark is detected (logic-1) or it is not (logic-0).

On the other hand, in multiple bit watermarking methods string of bits $b_1, b_2, \ldots b_M$ are embedded to an image. They first divide the image into M sub-images $I_1, I_2, \ldots I_M$ of size $m \times n$. Then a random watermark pattern having the same size as the sub-image, is added to each sub-image I_i. This is done by modulating the pattern according to the corresponding bit value b_i. The modulation can be done in several different ways. The simplest way is adding the random pattern of size $m \times n$ to the sub-image if the watermark bit is 1, and leaving the sub-image unchanged if the watermark bit is -1. Another method for modulating watermark bits is generating a pseudorandom pattern of $-1, 1$ for each bit of the watermark to be embedded. This pattern generation is similar to the sequence generation method used in Code Division Multiple Access (CDMA) spread spectrum. In this technique, if the watermark is $b_1, b_2, \ldots, b_M$, then M stochastically independent pseudorandom patterns $v1v2vM$ having the same size as the image are created. Each pattern, vi, is modulated by its corresponding bit, bi. The sum of all random patterns vi constructs the watermark. To apply this technique to 2D images, we replace the image with the $m \times n$ blocks and the watermark vectors with random -1 and 1.

2.5 Attacks on Watermarking

Many operations may affect the watermarking algorithms and destroy it. Those operations that destroy watermark data are called attacks. Image, audio, and video multimedia is generally stored in lossy compressed format. These compressions separate important and unimportant parts of data and discard the unimportant parts. This distortion may cause damage to watermark data too. Therefore, a simple attack is compressing multimedia data in a lossy way and destroying watermark. In case of image multimedia, a rotation or scaling can change pixel values and damage watermark, while preserving the visual content of the image. Signal processing operations such as quantization, decompression, re-sampling, color reduction, swapping some pixels and so on can damage the watermark data. Adding noise can also affect the inserted watermark data.

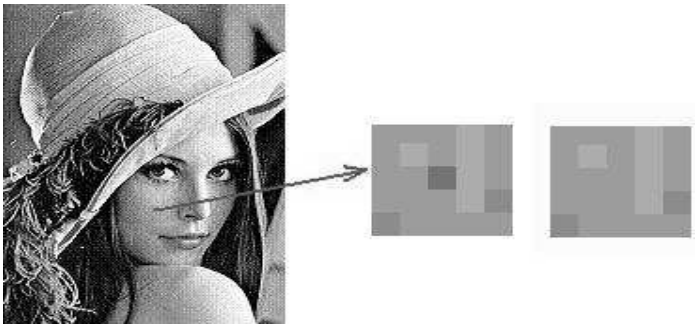

Figure 2.4: A description of the median filter with a window size of 5×5.

In general, different attacks on watermarking can be divided into two groups, namely unintentional and intentional attacks. Unintentional attacks happen with using normal signal processing operations such as compression, transcoding, printing/scanning, filtering, noise, geometric transforms, and cropping. For example, multimedia data is generally stored in lossy compressed format in order to use less storage capacities. These compression algorithms discard the unimportant parts of data. This distortion may cause damage of inserted watermark data too. This means that a simple attack is compressing multimedia data in a lossy way. In addition, a rotation or scaling can change pixel values and destroy the watermark data. Signal processing operations such as quantization, decompression, re-sampling, color reduction, can damage the watermark. For intentional attacks, a person on purpose can attack on inserted watermark data in order to copy the multimedia data.

Some attacks such as median filter, sharpening, and Gaussian filtering are explained using an example. Median filter sorts all values located in a window and picks the median value and replaces the value at the center of the window. Figure 2.4 depicts an example of this attack.

Sharpening is a high pass filter that amplifies high frequency contents of an image as is shown in the right image of Figure 2.5. On the other hand, Gaussian filtering is a low pass filter that reduces high frequency contents of an image. The result of the Gaussian filtering is depicted in the right image in Figure 2.6.

2.6 Hardware Solution

A watermark algorithm can be implemented using either software or hardware. In a software implementation, a watermark algorithm is performed as a code on a processor. The

Figure 2.5: Sharpening attack increases the high frequency values.

Figure 2.6: Gaussian filtering reduces high frequency values of an image

software implementations are flexible, while the software designers do not have control on hardware resources. On the other hand, in a hardware implementation, a watermark algorithm is fully performed in custom-designed hardware. A hardware implementation consumes less area and less power compared to a software implementation. Implementing a watermark algorithm using software solutions are usually faster than hardware implementations. However, hardware implementation is some cases are considered. For instance, in some embedded devices, a hardware watermarking solution is often more economical because it takes a small dedicated area. On the other hand, in software

watermarking solution, it needs a dedicated processor such as DSP that consumes more area and more power than the hardware solution [22].

In [22], a hardware for Just Another Watermarking System (JAWS) has been designed and implemented. The JAWS has also been implemented using a software solution by Trimedia DSP platform in [29]. The JAWS embeds some payload data into individual frames of video in spatial domain. The JAWS algorithm has the following stages. First, it calculates base watermark that computes a noise pattern from a set of n basic pseudo-random noise patterns and the key carriers. The set of basic patterns are fixed. The key is the user information. Second, it calculates embedding depth. The generated pseudo-random noise patterns are embedded to a certain depth. This embedding depth determines how much the luminance values of the frame pixels will change. Third, in order to generate the final watermark data, the outputs of the first and second stages are pixel-wise multiplied. Finally, the final watermark data is pixel-wise added with the original video frame to generate the watermarked frames.

The hardware architecture of the JAWS algorithm has the same stages as the software solution. The hardware was implemented as pipeline architecture, where each stage completes one operation and submits its results to the next stage. The first stage is calculating the base watermark data. This stage uses a precomputed random noise pattern, which are stored in a lookup table. The precomputed patterns are inserted to frame data. The second stage is calculating the embedded depths. This stage uses a high-pass filter to calculate the embedding depth. The high-pass filter is a 3×3 array. The filter operation needs at least 3 consecutive rows of frame data. This means that this stage needs a buffer to store three rows of a frame. When three rows are available in the buffer, they are filtered by the provided high-pass coefficients. Each filtered element needs nine pixel values read from the buffer. The third part multiplies the output results of the previous stages by each other and finally, the watermark data is added to the original frame pixels.

An energy efficient watermarking algorithm for mobile devices using proxy-based partitioning has been proposed in [13]. Their proposed system has three elements, namely, mobile devices, proxy servers, and content servers. Mobile devices could be any handheld devices such as Personal Digital Assistants (PDAs). Multimedia data are stored in the content servers. These servers send the requested data to the customers through the proxy servers. Proxy servers are intermediate resources between the mobile devices and content servers. The proxy servers can compress, decompress, multimedia data in real-time. Kejariwal et al. [13] have presented a partitioning watermarking algorithm between the proxy server an the customer devices. First, PADs devices send their watermark data to the proxy server, after that, the proxy servers execute whole watermark algorithm in order to insert the secret data to the original data.

Transform Domain Watermarking Algorithms

3

Two processing-domain categories have been proposed for digital watermarking algorithms, namely spatial domain and frequency domain. A watermark technique based on the spatial domain spread watermark data to be embedded in the pixel value. In other words, this approach uses minor changes in the pixel value intensity. The simplest example of the former approach is to embed the watermark in the least significant bit of image pixels [32]. Significant portions of low frequency componenets of images should be modified in order to insert the watermark data in a reliable and robust way. As another example, an image is divided into the same size of blocks and a certain watermark data is added with the sub-blocks [35]. However, the techniques in the spatial domain still have relatively low-bit capacity and are not resistant enough to lossy image compression and other image processing [11]. For instance, a simple noise in the image may eliminate the watermark. On the other hand, frequency domain-based techniques can embed more bits for watermark and are more robust to attack. In addition, transform-domain watermarking techniques are typically much more robust to image manipulation compared to the spatial domain techniques. This is because the transform domain does not use the original image for embedding the watermark data. Hence, in this chapter we discuss the important transforms such as DCT and DWT.

This chapter is organized as follows. In Section 3.1 we discuss the concept of the transform domain watermark insertion and watermark detection. We explain the DCT and DWT in Section 3.2 and in Section 3.3, respectively.

3.1 Transform Domain Watermark Insertion and Detection

Figure 3.1 depicts the block diagram for transform domain watermark insertion. First, the input image is transformed using a transform such as the DWT or DCT. In general, any frequency domain transform can be used. The watermark data is embedded to a transformed image. In other words, the watermark data is inserted into transformed coefficients. Finally, inverse transform is performed on the transformed watermarked image.

The watermark detection process is the inverse procedure of the watermark insertion process as is depicted in Figure 3.2. As this figure shows, to extract the watermark data from the watermarked image, first, the watermarked and the original image are transformed using the DCT or DWT. Second, the transformed image is subtracted from the transformed watermarked image. This is because the watermark data is the difference between the original image and the watermarked image. Finally, the similarity of the

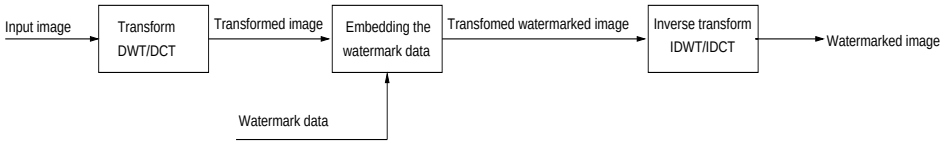

Figure 3.1: Block diagram of the transform domain watermark insertion.

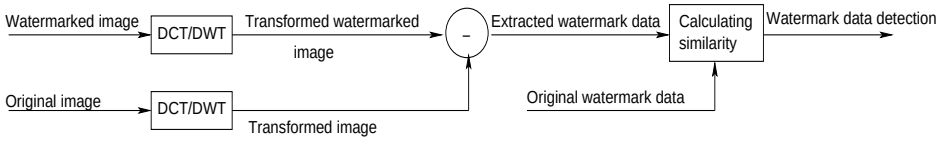

Figure 3.2: Block diagram of the transform domain watermark detection.

original watermark data and the extracted watermark data is computed. The similarity is depends on the amount of the inserted watermark data and the watermark attacks. Since the DCT and DWT are usually used for watermarking in the frequency domain, we explain these transformed in more detail in the following sections.

3.2 Discrete Cosine Transform

The transform of a signal is just another form of representing the signal. It does not change the information content present in the signal. Discrete cosine transform is widely used in image and video compression applications such as JPEG and MPEG. These multimedia standards partition an input image into 8×8 blocks after that the DCT for each block is computed. The watermarking techniques embed watermarking data into the middle frequency bands of a transformed image. The middle frequency bands are chosen such that they avoid the most visual parts of the image (the low frequencies) without overexposing themselves to removal through compression and noise attacks (high frequencies).

A 2D DCT is efficiently computed by 1D transforms on each row followed by 1D transforms on each column. There are different algorithms to compute the 2D DCT. For example, one such algorithm is by using matrix multiplication and it is explained as follows. The $M \times M$ transform matrix T is given by Equation (3.1):

$$T_{ij} = \begin{cases} \frac{1}{\sqrt{M}}, & \text{if } i = 0, 0 \leq j \leq M-1. \\ \sqrt{\frac{2}{M}} * \cos \frac{\pi(2j+1)i}{2M} & \text{if } 1 \leq i \leq M-1, 0 \leq j \leq M-1. \end{cases} \tag{3.1}$$

For an $M \times M$ matrix A, $T * A$ is an MxM matrix whose columns contain the 1D DCT of the columns of A. The 2D DCT of A can be computed as $B = T * A * T^{'}$.

Another technique to compute the 2D DCT is using Loeffler, Ligtenberg, and Moschytz (LLM) technique [19]. This algorithm has four stages, the output of each

```
A0 = x0 + x7;
A1 = x1 + x6;
A2 = x2 + x5;
A3 = x3 + x4;
A4 = x3 - x4;
A5 = x2 - x5;
A6 = x1 - x6;
A7 = x0 - x7;
```

Figure 3.3: First stage of LLM algorithm for implementation of the 2D DCT.

stage is sent to next stage. The first step, for example has four addition and four subtraction operations. Figure 3.3 depicts the C implementation of the the first stage of the LLM [19] algorithm to implement the 2D DCT for a block of 8×8.

The JPEG2000 and MPEG-4 multimedia standards employ the DWT instead of DCT that is used in older multimedia standards. The main reason is that higher image quality can be achieved at lower bit rates. In addition, the DCT based compression standards are block-based causing blocking artifacts in the output image. The DWT avoids the artifact problem by operating on a complete image or a large part of an image. Consequently, it requires more memory than the DCT.

3.3 Discrete Wavelet Transform

In this section, we give a brief explanation of the DWT.

3.3.1 1D Discrete Wavelet Transform

Wavelet can represent a signal in time-frequency domain. Analyzing a signal with this kind of representation gives more information about the when and where of different frequency components. In other words, wavelet transform not only transforms the time domain representation of a signal to the frequency domain representation, but also preserves spatial information in the transform. This feature enhances the image quality especially for the low bit rate representation. The DWT is a multi-resolution technique that can analyze different frequencies by different resolutions.

The low-pass and high-pass filter pair is known as *analysis filter-bank*. An example of a low-pass filter is $h_0(n) = (-1, 2, 6, 2, -1)/8$, which is symmetric and has five integer coefficients. An example of a high-pass filter is $h_1(n) = (-1, 2, -1)/2$, which is also symmetric and has three integer coefficients. These low- and high pass filters are used in the $(5, 3)$ filter transform.

After applying the 1D DWT on a signal that has been decomposed into two bands, the low-pass outputs are still highly correlated, and can be subjected to another stage of two-band decomposition to achieve additional decorrelation. In addition, the 1D DWT

can be easily extended to two dimensions (2D) by applying the filter-bank in a separable manner. At each level of the wavelet decomposition, each row of a 2D image is first transformed using a 1D horizontal analysis filter-bank (h_0, h_1). The same filter-bank is then applied vertically to each column of the filtered and sub-sampled data.

3.3.2 2D Discrete Wavelet Transform

The 2D DWT is computed by performing low-pass and high-pass filtering of the image pixels as shown in Figure 3.4. In this figure, the low-pass and high-pass filters are denoted by h and g, respectively. This figure depicts the three levels of the 2D DWT decomposition. At each level, the high-pass filter generates detailed image pixels information, while the low-pass filter produces the coarse approximations of the input image. At the end of each low-pass and high-pass filtering, the outputs are down-sampled by two ($\downarrow 2$). In order to provide 2D DWT, 1D DWT is applied twice in both horizontal and vertical filtering. In other words, a 2D DWT can be performed by first performing a 1D DWT on each row, which is referred to as horizontal filtering, of the image followed by a 1D DWT on each column, which is called vertical filtering.

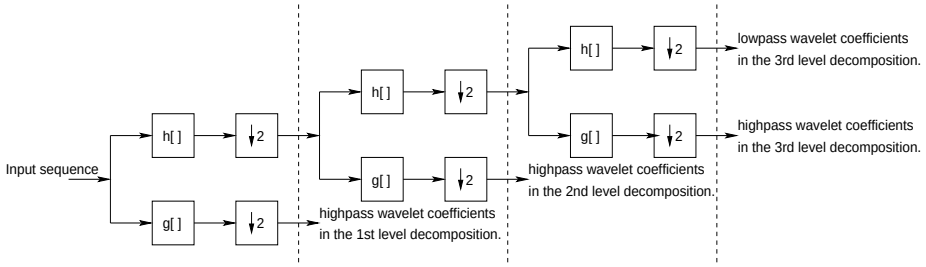

Figure 3.4: Three level 2D DWT decomposition of an input image using filtering approach. The h and g variables denote the low-pass and high-pass filters, respectively. The notation of ($\downarrow 2$) refers to down-sapling of the output coefficients by two.

Figure 3.5 illustrates the first decomposition level ($d = 1$). In this level the original image is decomposed into four sub-bands that carry the frequency information in both the horizontal and vertical directions. In order to form multiple decomposition levels, the algorithm is applied recursively to the LL sub-band. Figure 3.6 illustrates the second ($d = 2$) and third ($d = 3$) decomposition levels as well as the layout of the different bands.

There are different approaches to implement 2D DWT such as traditional convolution-based and lifting scheme methods. The convolutional methods apply filtering by multiplying the filter coefficients with the input samples and accumulating the results. Their implementation is similar to the Finite Impulse Response (FIR) implementation.This kind of implementation needs a large number of computations.

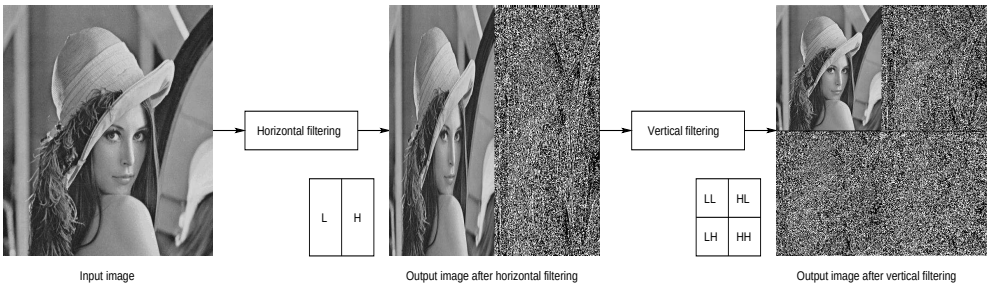

Figure 3.5: Different sub-bands after first decomposition level.

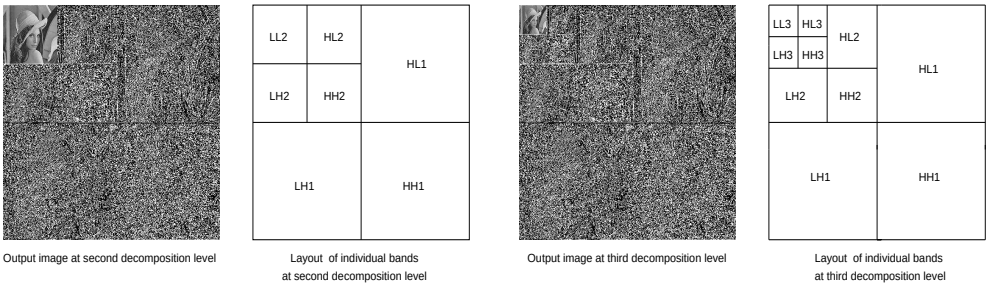

Figure 3.6: Sub-bands after second and third decomposition levels.

3.3.3 Lifting Scheme

The lifting scheme has been proposed for the efficient implementation of the 2D DWT. The lifting approaches need 50% less computations than the FIR approaches. The basic idea of the lifting scheme is to use the correlation in the image pixels values to remove the redundancy [7, 10]. The lifting scheme has three phases, namely, split, predict, and update, as illustrated in Figure 3.7. In the split stage, the input sequence is split into

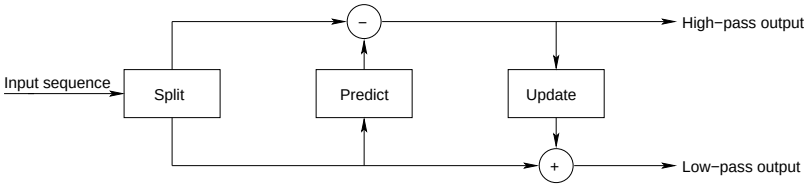

Figure 3.7: Three different phases in the lifting scheme.

two sub-sequences consisting of the even and odd samples. In the predict and update stages, the high-pass and low-pass values are computed, respectively. Lifting scheme has many advantages than traditional convolution-based algorithms. Some of them are following [7, 10]:

- Lifting scheme algorithms provide more performance improvement than traditional

convolution-based implementation. The number of operation in the lifting scheme approaches is almost one half of that in the convolution-based approaches. Such reduction in the computational complexity makes lifting scheme filters attractive for high-performance implementation.

- In lifting scheme implementation, there is no need to manage the borders of an image, while in the traditional convolution-based approaches it is necessary.
- Lifting scheme approaches can be implemented in-place. This means that the DWT can be processed without using auxiliary array as is necessary in the traditional convolution-based implementation.
- All operations within one lifting step can almost be performed in parallel.
- With lifting transforms it is easy to build non linear wavelet transforms. For instance, wavelet transforms that map integers to integers. Such transformations are more important for hardware implementation and for lossless image compression.
- The inverse transform of lifting scheme is straightforward, it easily inverts the order of functionality. So the same operations and resources could be reused to implement IDWT.

One example of this group is the integer-to-integer $(5,3)$ lifting scheme ($(5,3)$ *lifting*). The resulting forward transform for the $(5,3)$ lifting that has been used is given by Equation (3.2) and Equation (3.3).

$$y(2n+1) = x(2n+1) - \lfloor \frac{x(2n)+x(2n+2)}{2} \rfloor. \tag{3.2}$$

$$y(2n) = x(2n) + \lfloor \frac{y(2n-1)+y(2n+1)+2}{4} \rfloor. \tag{3.3}$$

Figure 3.8 depicts the C implementation of the first level 2D DWT decomposition using $(5,3)$ lifting transform for an $N \times M$ image.

3.3.4 Convolutional Methods

The convolutional methods apply filtering by multiplying the filter coefficients with the input samples and accumulating the results. The Daubechies' transform with four coefficients [31] (Daub-4) and the Cohen, Daubechies and Feauveau 9/7 filter [6] (CDF-9/7) are examples of this category. For instance, the CDF-9/7 transform has 9 low-pass filter coefficients $h = \{h_{-4}, h_{-3}, h_{-2}, h_{-1}, h_0, h_1, h_2, h_3, h_4\}$ and 7 high-pass filter coefficients $g = \{g_{-2}, g_{-1}, g_0, g_1, g_2, g_3, g_4\}$. Both filters are symmetric, i.e., $h_{-i} = h_i$.

Figure 3.9 depicts the C implementation of the first level 2D DWT decomposition using the Daub-4 transform for an $N \times M$ image [27].

Figure 3.10 depicts the C implementation of the first level 2D DWT decomposition using the CDF-9/7 transform for an $N \times M$ image [30].

```
#define    N    512
#define    M    N
int            i,j,jj;
int  in_image[N+2][M], ou_image[N+2][M];

void firstleveldwt_53() {
for (i=0; i<N; i++)
   for (j=1, jj=0; j<M; jj++, j +=2)   {
   // Calculation of high-value
      ou_image[i][jj + M/2] = in_image[i][j]-((in_image[i][j-1]+in_image[i][j+1]) >> 1)
  // Calculation of low value
      ou_image[i][jj] = in_image[i][j-1]+((ou_image[i][jj + half_col]+ou_image[i][jj+
      M/2 - 1] + 2) >> 2);
}
for (j=1, jj=0; j<N; jj++, j +=2)
   for (i=0; i<M; i++)  {
   // Calculation of high-value
      in_image[jj + N/2][i]=ou_image[j][i]-((ou_image[j-1][i]+ou_image[j+1][i]) >> 1);
  //calculation of low value
     in_image[jj][i]=ou_image[j-1][i]+((in_image[jj + N/2][i] + in_image[jj + N/2 -1][i]+
     2) >> 2);\index{}
   }
}
```

Figure 3.8: C implementation of the first level 2D DWT decomposition using $(5, 3)$ lifting transform for an $N \times M$ image.

```
void first levelDaub_4() {
{
   int i, j, jj;
   float low[] ={-0.1294, 0.2241, 0.8365 , 0.4830};
   float high[]={-0.4830, 0.8365, -0.2241, -0.1294};
   float in_image[N+3][M], ou_image[N+3][M];

for (i=0; i<N; i++)
   for(j=0, jj=0; jj<M; j++, jj +=2)   {
    ou_image[i][j]=in_image[i][jj]* low[0]+ in_image[i][jj+1]*low[1]+
    in_image[i][jj+2]*low[2]+in_image[i][jj+3]*low[3];

    ou_image[i][j+M/2]=in_image[i][jj]* high[0]+in_image[i][jj+1]*high[1]+
             in_image[i][jj+2]*high[2]+in_image[i][jj+3]*high[3];
   }
for (i=0,  jj=0; jj<N; i++, jj +=2)
   for(j=0; j<M; j++)   {
      in_image[i][j]=ou_image[jj][j]* low[0]+ou_image[jj+1][j]* low[1]+
      ou_image[jj+2][j]*low[2]+ou_image[jj+3][j]* low[3];
       in_image[i + N/2][j]=ou_image[jj][j]* high[0]+ou_image[jj+1][j]* high[1]+
       ou_image[jj+2][j]*high[2]+ou_image[jj+3][j]*high[3];
      }
}
```

Figure 3.9: C implementation of the first level 2D DWT decomposition using the Daub-4 transform for an $N \times M$ image.

```
void first levelCDF97() {
{
 int i, j, jj;
 float in_image[N+8][M], ou_image[N+8][M];
 float low[] ={0.6029499018236360,0.266864118442875,-0.078223266528990,
                    -0.016864118442875, 0.026748757410810};
 float high[]={-0.591271763114250, -0.057543526228500,
                      0.091271763114250, 1.115087052457000};
 for (i=0; i<N; i++)
   for(j=0, jj=4; jj<M - 4; j++, jj +=2) {
     ou_image[i][j] =in_image[i][jj - 4] * low[4]+in_image[i][jj - 3] * low[3]+
     in_image[i][jj -2]*low[2]+ in_image[i][jj - ]*low[1]+
     in_image[i][jj]* low[0] +  in_image[i][jj + ]* low[1]+ in_image[i][jj + 2]*low[2];
     in_image[i][jj+ 3]* low[3]+ in_image[i][jj+ 4]* low[4];

     ou_image[i][j + M/2]=in_image[i][jj - 4]* high[2] + in_image[i][jj - 3] * high[1]+
     in_image[i][jj -2]*high[0]+in_image[i][jj - 1] * high[3]+
     in_image[i][jj]*high[0]+ in_image[i][jj+1]* high[1]+ in_image[i][jj+2]* high[2];
  }
for (i=0,  jj=4; jj<N - 4; i++, jj +=2)
   for(j=0; j<M; j++)  {
     in_image[i][j]= ou_image[jj - 4][j]* low[4] + ou_image[jj - 3][j]* low[3]+
     ou_image[jj - 2][j]* low[2]+ou_image[jj - 1][j]* low[1]+
     ou_image[jj][j]* low[0] + ou_image[jj + 1][j]*low[1]+
     ou_image[jj + 2][j] * low[2] +
     ou_image[jj + 3][j] * low[3]+ou_image[jj+4][j]* low[4];

     in_image[i + N/2][j]= ou_image[jj - 4][j]*high[2]+ou_image[jj -3][j]*high[1]+
     ou_image[jj - 2][j] * high[0]+ou_image[jj - 1][j] * high[3] +
     ou_image[jj][j]* high[0]+ ou_image[jj+1][j] * high[1] +
     ou_image[jj+2][j]* high[2];
   }
}
```

Figure 3.10: C implementation of the first level 2D DWT decomposition using the CDF-97 transform for an $N \times M$ image.

3.3.5 2×2 Haar Transform

The 2×2 Haar transform is used in image compression [36]. The 2×2 Haar transform is sometimes referred to as a wavelet. A 2D Haar transform can be performed by first performing a 1D Haar transform on each row (horizontal Haar transform) followed by a 1D Haar transform on each column (vertical Haar transform). This transform is used to decompose an image into four different bands. The 1D 2×2 Haar transform replaces adjacent pixel values with their sums and differences. Figure 3.11 depicts an example of the 2D 2×2 Haar transform for a 4×4 image. This transform generates four different subbands of low-pass and high-pass values.

Figure 3.12 depicts the C implementation of the first level 2D DWT decomposition using the $\mathbf{2 \times 2}$ Haar transform for an $N \times M$ image.

Haar DWT has been used in [36, 18]. The second level 2D DWT decomposition has been obtained using Haar transform.

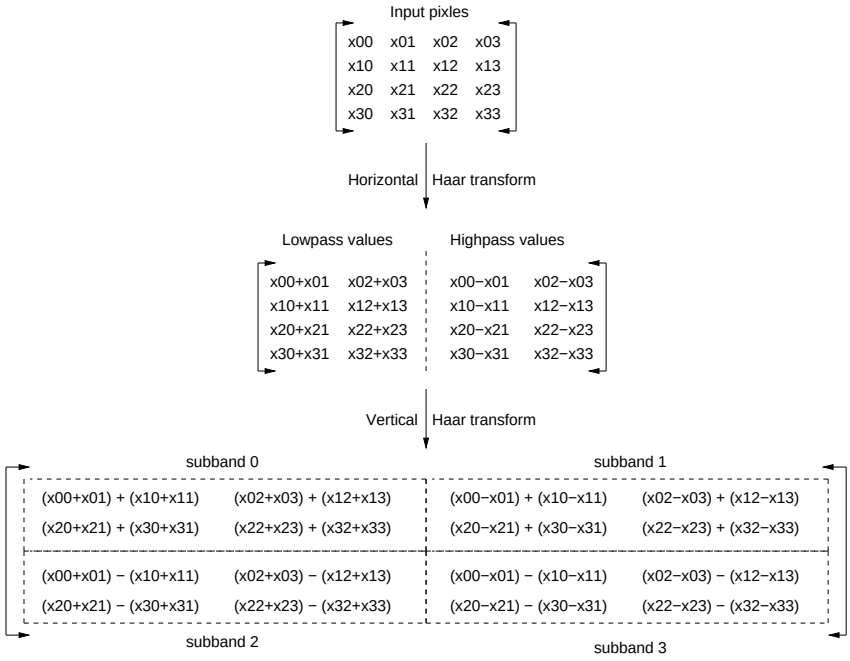

Figure 3.11: 2D 2×2 Haar transform using two 1D horizontal and vertical Haar transform.

```
void HaarTransform(void)
{
int       i, j, ii, jj;
char      BlockInput[N][M];
short     BlockOutput[N][M];
  for (i=0, ii=0; i<N; ii++, i +=2)
    for (j=0, jj=0; j<M; jj++, j +=2) {
      BlockOutput[ii][jj]=(BlockInput[i][j]+BlockInput[i][j+1])+
      (BlockInput[i+1][j]+BlockInput[i+1][j+1]);

      BlockOutput[ii][jj+ M/2]=(BlockInput[i][j]+BlockInput[i][j+1])-
      (BlockInput[i+1][j] + BlockInput[i+1][j+1]);

      BlockOutput[ii+N/2][jj]=(BlockInput[i][j]-BlockInput[i][j+1])+
      (BlockInput[i+1][j] - BlockInput[i+1][j+1]);

      BlockOutput[ii+N/2][jj + M/2]=(BlockInput[i][j]-BlockInput[i][j+1])-
      (BlockInput[i+1][j] - BlockInput[i+1][j+1]);
    }
}
```

Figure 3.12: C implementation of the first level 2D DWT decomposition using the $\mathbf{2 \times 2}$ Haar transform for an $N \times M$ image.

There are different algorithms to traverse an image to implement these transforms, namely *Row-Column Wavelet Transform* (RCWT) and *Line-Based Wavelet Transform*

(LBWT) [1, 2, 3, 4, 5]. These approaches are discussed in the following sections.

3.3.6 Row-Column Wavelet Transform

In the RCWT approach, the 2D DWT is divided into two 1D DWTs, namely horizontal and vertical filtering. Horizontal filtering processes the rows of the original image and stores the wavelet coefficients in an auxiliary matrix. Thereafter, the vertical filtering phase processes the columns of the auxiliary matrix and stores the results back in the original matrix. In other words, this algorithm requires that all lines are horizontally filtered before the vertical filtering starts. The computational complexity of both horizontal and vertical filtering is the same. Figure 3.5 depicts both horizontal and vertical filtering. Each of these filtering is applied separately. Each of these $N \times M$ matrices requires NMc bytes of memory, where c denotes the number of bytes required to represent one wavelet coefficient. The RCWT traversal technique has been used to implement the 2D DWT in this chapter.

3.3.7 Line-Based Wavelet Transform

In the line-based wavelet transform approach, the vertical filtering starts as soon as a sufficient number of lines, as determined by the filter length, has been horizontally filtered. In other words, the LBWT algorithm uses a single loop to process both rows and columns together. This technique computes the 2D DWT of an $N \times M$ image by the following stages. First, the horizontal filtering filters L rows, where L is the filter length, and stores the low-pass and high-pass values interleaved in an $L \times M$ buffer. Thereafter, the columns of this small buffer are filtered. This produces two wavelet coefficients rows, which are stored in different subbands in an auxiliary matrix in the order expected by the quantization step. Finally, these stages are repeated to process all rows and columns. Figure 3.13 illustrates the LBWT algorithm.

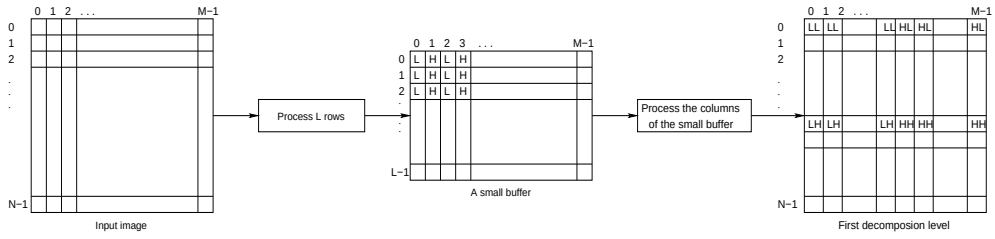

Figure 3.13: The line-based wavelet transform approach processes both rows and columns in a single loop.

Figure 3.14 represents a part of the C implementation of the main part, for the Daub-4 transform. As this figure depicts, there is an outer loop with index i. Inside this loop, there are two inner loops. The first inner loop with index j is related to the horizontal filtering on the input matrix, and the other inner loop with also index j is related to the vertical filtering on the calculated results from previous loop. In each

```
void Both_Hori_Ver()
{
   int i, j, jj;
   float low[] ={-0.1294, 0.2241, 0.8365 , 0.4830};
   float high[]={-0.4830, 0.8365, -0.2241, -0.1294};
   for (i=0, ii=0; i<N; ii++, i +=2)
   {
     k = ( ii % 2 ) * 2;
     for(j=0, jj=0; jj<M; j++, jj +=2) {
       BufLow[k][jj]  = in_image[i][jj] * low[0]   + in_image[i][jj + 1] * low[1]  +
                              in_image[i][jj + 2] * low[2] + in_image[i][jj + 3] * low[3];
       BufLow[k][jj + 1  = in_image[i][jj] * high[0]  + in_image[i][jj + 1]* high[1] +
                           in_image[i][jj + 2]* high[2]+ in_image[i][jj + 3]* high[3];

       BufLow[k+1][jj] = in_image[i+1][jj] * low[0] +  in_image[i+1][jj + 1]* low[1] +
                   in_image[i+1][jj + 2] * low[2]+ in_image[i+1][jj + 3] * low[3];
       BufLow[k+1][jj + 1] = in_image[i+1][jj] * high[0] + in_image[i+1][jj  + 1] * high[1] +
                   in_image[i+1][jj + 2] * high[2] + in_image[i+1][jj + 3]* high[3];
     }
     for(j=0, jj=0; jj<M; j++, jj +=2)
     {
       ou_image[ii][j] = BufLow[0][jj]   * low[0]  +  BufLow[1][jj] * low[1]+
                              BufLow[2][jj]   * low[2]  +  BufLow[3][jj] * low[3];
       ou_image[ii + N/2][j] = BufLow[0][jj]   * high[0] +  BufLow[1][jj] * high[1] +
                         BufLow[2][jj]* high[2] +  BufLow[3][jj] * high[3];

       ou_image[ii][j + M/2]  = BufLow[0][jj + 1] * low[0] + BufLow[1][jj + 1]* low[1] +
                             BufLow[2][jj+1]* low[2]+ BufLow[3][jj + 1]* low[3];

       ou_image[ii + N/2][j + M/2] = BufLow[0][jj + 1] * high[0] + BufLow[1][jj+1]* high[1] +
                        BufLow[2][jj+1]*high[2] + BufLow[2][jj+1]* high[3];
     }
   }
}
```

Figure 3.14: A part of the C implementation of the line-based wavelet transform algorithm for the Daub-4 transform.

iteration, the horizontal filtering processes two consecutive image rows, and it passes four rows of calculated wavelet coefficients to the vertical filtering [26].

Watermark Algorithms Based on DCT and DWT

4

In this chapter we discuss two and six watermark algorithms, which have been proposed based on the DCT and DWT, respectively. We focus more on wavelet-domain watermarking algorithms than on DCT. The reasons for this are as follows. First, some multimedia standards such as JPEG2000 and MPEG-4 are based on DWT. These new standards brought new requirements such as progressive and low bit rate transmission as well as region-of-interest coding. In other words, this approach matches the emerging image and video compression standards. Second, watermarking technique in the DWT domain has multi-resolution characteristics and is hierarchical. When the received image is not significantly damaged, the cross correlations with the whole size of the image may not be necessary, and therefore, much of the computational load can be saved. Finally, it is usually true that the human eyes are not sensitive to the small changes in image edges (high frequencies), while they are very sensitive to small changes in the smooth parts of an image (low frequencies). In the DWT domain, the image edges are usually well defined to the high frequency sub-bands, for instance HH, LH, HL parts. Large coefficients in these bands usually indicate edges in an image. Therefore, adding watermarks to these large wavelet coefficients is difficult for the human eye to perceive [36].

This capter is organized as follows. In Section 4.1 we discuss two watermarking algorithms using DCT. We also explain six watermarking algorithms using DWT in Section 4.2.

4.1 Watermark Algorithms using the DCT

In this section two watermark algorithms, which have been proposed using the DCT transform in [12, 15] are discussed.

4.1.1 Cox's Algorithm

Cox et al. [12] proposed spread spectrum watermarking algorithm. In this technique, the watermark data is spread over many frequency values so that the energy in any value is very small and undetectable. In order to implement this algorithm, a sequence of values $V = v_1, v_2, ..., v_n$ are extricated from each image I. Then the watermark data $X = x_1, x_2, ..., x_n$ are inserted into the extracted values V, to obtain a sequence of $V' = v'_1, v'_2, \cdots, v'_n$ using a scaling factor α with the following equations.

$$v_i' = v_i + \alpha x_i. \quad (4.1)$$
$$v_i' = v_i(1 + \alpha x_i). \quad (4.2)$$

In order to obtain the final watermarked image I' , the sequence of V' is inserted into the original image I.

In the above equations, one single scaling parameter α has been chosen and it may not be applicable for all image pixels. Generally, it is possible to have multiple scaling parameters $\alpha_1, \alpha_2, \alpha_3, ..., \alpha_n$, however, selecting multiple scaling values is difficult. In [12], Equation (4.2) with a single parameter $\alpha = 0.1$ has been used. Watermark data has been constructed as an independent and identically distributed Gaussian random vector. In other words, the watermark data consists of random numbers drawn from an $N(0, 1)$ distribution.

Watermark data has been placed in the most significant coefficients of the transformed image. They added the watermark data to the 1000 largest coefficients of the DCT. This operation maximizes the chances of detecting the watermark even after common signal processing operations. For color images, first, those images are converted to black and white images using $YCbCr$ representation. Second, the brightness values Y are watermarked. The watermarked image can be converted to other formats, but must be converted back to $YCbCr$ representation to extract the watermark data. In other words, color images are robust to the signal processing operations by applying the watermark algorithms to gray level images.

In the Cox's algorithm an original image is used for detecting watermark. Watermark data should not be placed in a perceptually insignificant region of the image since many common signal and geometric processes affect these components. For instance, inserting watermark in a high frequency spectrum of an image can damage or change by any process that directly or indirectly performs low-pass filtering. Hence, the problem to be solved in this algorithm is insertion of watermark data into the most perceptually significant regions of the spectrum.

4.1.2 Koch's Algorithm

Koch et al. [15] proposed a watermark algorithm using the DCT domain. First, they transformed an image using the DCT transform and then pseudo-random numbers are inserted into a subset of blocks. A triblet of blocks with mid-range frequencies was slightly revised to obtain a binary sequence watermark. They discussed the Randomly Sequenced Pulse Position Modulated Code (RSPPMC) technique. This algorithm consists of two parts. The first part produces the actual copyright code and a random sequence of locations for embedding the code in the image. The second part embeds the code at the specified locations using a simple pulsing technique. This part includes the following steps. First, the position sequence is used to generate a sequence of locations for mapping the pixels. Second, the blocks of image data are transformed and quantized. Third, the watermark data, code pulses, represents the binary code being embedded on

selected locations. Finally, the quantized data is decoded after the inverse transform of the watermarked transformed image is obtained.

4.2 Watermark Algorithms using the DWT

In this section, six watermark algorithms based on the DWT are discussed.

4.2.1 Xie's Algorithm

Xie et al. [37] has developed a blind watermark technique in the DWT domain. Xie's algorithm modified the wavelet coefficients using a median filter with a 1×3 sliding window. A non-overlapping 3×1 window runs through the entire low frequency band of the wavelet coefficients. For example, elements in the window are denoted as b_1, b_2, b_3 corresponding to the coordinates $(i-1, j), (i, j), (i+1, j)$, respectively. They are sorted as $b_{(1)} \leq b_{(2)} \leq b_{(3)}$. A nonlinear transformation based on the following equation is performed in oder to change the median of these coefficients, while the remaining coefficients do not change.

$$b_2' = f(\alpha, b_{(1)}, b_{(3)}, x). \tag{4.3}$$

x is the bit of the watermark being inserted in the location of the window. This transformation changes the median of a local area to a value set by its neighbors.

$$S_\alpha = \alpha \frac{|b_{(1)}| + |b_{(3)}|}{2}. \tag{4.4}$$

α is a parameter with a default value of 0.05. Then, the range of the coefficients $b_{(1)}$ and $b_{(3)}$ are partitioned into k intervals $(l_0, l_1, ..., l_k)$ so that each interval is in length of S_α. This means that $l_0 = b_{(1)}$ and $l_1 = b_{(1)} + S_\alpha$ and etc. Then $b'_{(2)}$ is defined as follows.

$$b'_{(2)} = l_m \quad \text{if m is odd and x = 1 or m is even and x = 0.} \tag{4.5}$$
$$= l_{m-1} \quad \text{if m is even and x = 1 or m is odd and x = 0.} \tag{4.6}$$

4.2.2 Xia's Algorithm

Xia et al. [36] insert pseudo-random codes to the large coefficients at the high and middle frequency bands of the DWT for an image. The idea is the same as the spread spectrum watermarking idea proposed by Cox et al. [12]. A pseudo-random sequence, Gaussian noise sequence $N[m, n]$ with mean 0 and variance 1 are inserted to the largest wavelet coefficients. Wavelet coefficients at the lowest resolution are not changed. In other words,

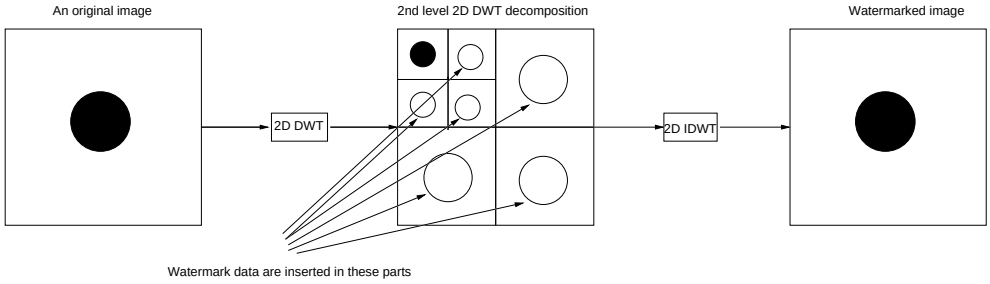

Figure 4.1: Inserting the watermark data into the largest wavelet coefficients in the second level 2D DWT decomposion.

Xia embedded the watermark data to all sub-bands except LL sub-band. This operation is shown in the following equation.

$$\bar{y}[m,n] = y[m,n] + \alpha y^2[m,n]N[m,n]. \tag{4.7}$$

where $y[m,n]$ denotes the wavelet coefficients, α is a parameter to control the level of the watermark, y^2 $[m,n]$ indicates the largest wavelet coefficients. After that, the 2D IDWT of $\bar{y}$ is computed. These steps are depicted in Figure 4.1.

As this figure shows the second level 2D DWT decomposition has been obtained. In other words, the image is decomposed into seven sub-bands. Gaussian noises are added into all six sub-bands except the lowest sub-band.

4.2.3 Wang's Algorithm

A watermark techniue based on DWT has been proposed by Wang et al. [34]. They search significant wavelet coefficients in different sub-bands to embed the watermark data. The searched significant coefficients are sorted according to their perceptual importance. The watermark data is adaptively weighted in different sub-bands to achieve robustness. This algorithm has the following stages.

1. The initial threshold T_s of each sub-band is set to one half of its maximum absolute value of wavelet coefficients inside the sub-band.

2. A sub-band with the maximum value of $\beta_s T_s$ is selected, where β_s is the weighting factor of sub-band s. For the selected sub-band, we test all un-selected wavelet coefficients and choose wavelet coefficients greater than the current threshold T_s as significant wavelet coefficients.

3. The new threshold is updated using $T_S^{new} = T_s/2$.

4. Repeat steps 2 and 3 until all wavelet coefficients are traced.

4.2.4 Tsun's Algorithm

Tsun et al. [18] also proposed a watermarking algorithm using DWT. The proposed idea consists of the following steps.

- Performing n level the 2D DWT decomposition level, on the input image and generate the output image X.
- Generate a binary matrix A with the same dimension and structure of the matrix X using a secret key. The secret key is shared between the embedder and authenticator.
- Generating a binary matrix B with the same dimension as matrix X such that all its pixels corresponding to the non-zero-valued coefficients in X are set to 1 and the rest set to 0, while B_{HH1} is modified by projection operation on all bands at every level to this band. This projection is performed in the following manner.
  An OR operation is performed on all of three sub-bands, LL_i, LH_i, and HL_i at each level of B to generate a sequence of binary sequences as $C = \{C_n, C_{n-1},...,C_1\}$ according to the following equations.

$$C_n(i,j) = 1, if B_{HLn}(i,j) or B_{LHn}(i,j) or B_{HHn}(i,j) or B_{LLn}(i,j) = 1 \quad (4.8)$$
$$C_n(i,j) = 0, otherwise. \quad (4.9)$$

  and

$$C_l(i,j) = 1, if B_{HLn}(i,j) or B_{LHn}(i,j) or B_{HHn}(i,j) = 1 \quad (4.10)$$
$$C_l(i,j) = 0 otherwise for all l < n. \quad (4.11)$$

- Binary watermark sequence is calculated by XOR of matrix A and matrix B that is called matrix W, $W = A$ XOR B. In other words, a binary watermark W is generated by performing XOR on the matrix A and matrix B.
- Wavelet coefficients in sub-band X_{HH1} are watermarked that their counterparts in sub-band B_{HH1} have a value of 1.
- Watermark bit is embedded into watermarkable wavelet coefficient X_{HH1}(i,j).

4.2.5 Kim's Algorithm

In order to equally embed the watermark to the whole image, Kim et al. [14] embedded watermark data to all sub-bands. The watermark data has been generated using a Gaussian distributed random vector. Level-adaptive thresholding has been used in order to select significant coefficients for each sub-band and different decompositions. They used 3-level decompositions and the length of the watermark is about 1000. The flowchart of inserted watermark data using 2D DWT is depicted in Figure 4.2.

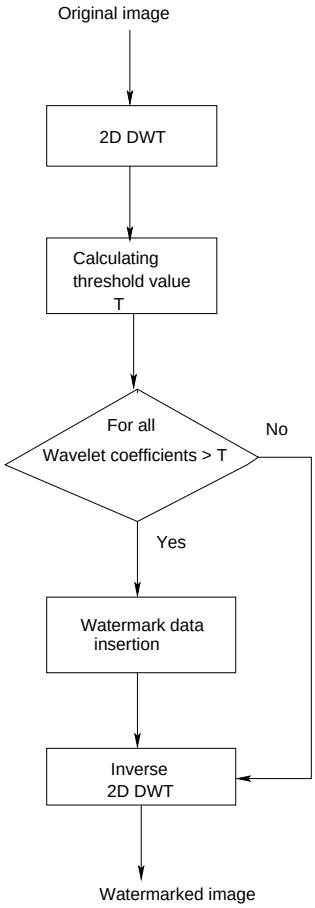

Figure 4.2: The flowchart of watermark data insertion for Kim's algorithm.

The watermark data is embedded using Equation (4.12).

$$v_i' = v_i + \alpha v_i x_i \tag{4.12}$$

where v_i is the selected wavelet coefficient and x_i is a gaussian distributed random vector. The scale factors of 0.4, 0.2, and 0.1 have been used for scale factor α, for decomposition levels 1, 2, and 3, respectively. In addition, v_i' denotes the DWT coefficient of the watermarked image.

Kim's algorithm selects the perceptually significant coefficients from wavelet coefficients using multi-level thresholding. After that, watermark data is embedded to selected coefficients. For invisibility and robustness of watermark, this algorithm uses Gaussian distributed random vector as the watermark.

4.2.6 Dugad's Algorithm

Dugad et al. [8] have inserted the watermark data to sub-bands, which have coefficients larger than a given threshold T_1 except the low-pass sub-bands. Picking all wavelet coefficients above a threshold is a natural way of adapting the amount of watermark added to the image. They used three 2D DWT decomposition levels. Robustness requires the watermark to be added in significant coefficients in the transform domain. However, the order and number of these significant coefficients can change due to various image manipulations. Adding watermark data to significant wavelet coefficients in the high frequency bands is equivalent to adding watermark to the edge areas of the image, which makes the watermark invisible to the human eye. Watermark data is embedded using Equation (4.12), the same as in the previous algorithm. x_i is generated from a uniform distribution of zero mean and unit variance and α is set to 0.2. However, this method does not require the original image for detecting the watermark data.

Watermark Robustness Evaluation

5

In this chapter we discuss and present our experimental results of watermark robustness evaluation. We have applied different attackes on the previous discussed watermarking algorithms. Generally speaking, the frequency based watermarking algorithms show a better performance compared to spatial based methods. Even very simple attacks such as median filtering or a lossy compression, can destroy the embedded watermark when spatial based methods are involved. Therefore, in this study only frequency based watermarking methods have been considered. The major consideration in frequency based methods is the robustness of the method to different attacks. To achieve a better performance with respect to robustness, the watermark should be embedded in the lower frequency contents of the image as far as possible. The reason is that the main watermark attacks change or eliminate high frequency content of the image. From this viewpoint, the main difference between the frequency based watermarking algorithms, is their choice of frequency coefficients to use for embedding watermark data. In fact the robustness of the algorithms is also dependent on the frequency at which the watermark data is added.

This chapter in organized as follows. We discuss two criteria for quality measurements of the original image and the watermarked image in Section 5.1. In Section 5.2, we discuss the images database that have been used as benchmark in our experimental results. We describe different watermark attacks in Section 5.3 and our experimental results about the evaluation of robustness against different attacks are presented in Section 5.4.

5.1 Quality Measurements

Two commonly measurements that are used to quantify the error between images are namely, Mean Square Error (MSR) and PSNR. Their equations are as follows.

$$MSE = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}(f(i,j)-g(i,j)). \tag{5.1}$$

$$PSNR = 10log_{10}\frac{255^2}{MSE}. \tag{5.2}$$

Where the sum over i and j denote the sum over all image pixels. Increasing PSNR represents increasing fidelity of compression. In general when the PSNR is 40 dB or larger, then the two images are virtually indistinguishable by human observers. In other words, the transformed image is almost identical to the original.

5.2 Images Database

It is important to evaluate an image watermark algorithm on many different images. Images should cover a broad range of contents and types. We have used 26 images provided at Fabien Petitcolas database [24] as a standard evaluation database for watermarking algorithms. Figure 5.1 indicates some of the sample images used in the experiments.

Figure 5.1: Sample images used for experiments.

We have performed evaluation by having in mind that the embedded watermark should be invisible so we have kept the PSNR value of the images constant at 35dB and compared the robustness of the different methods. The average PSNR and MSE values of test images versus watermarking algorithm used in our experiments are given in Table 5.1. The MSE and PSNR values are obtained from comparing original images with their watermarked correspondences. We have tried to keep PSNR close to 35 dB to avoid having a visible watermark but at the same time including the watermark with a large energy to be resistant to attacks.

Table 5.1: PSNR and MSE values for watermarked images of the experiments.

	TSUAN	COX	DUGAD	KIM	KOCH	WANG	XIA	XIE
PSNR	34.94	35.04	35.04	35.01	35.00	35.03	34.99	35.06
MSE	20.81	20.35	20.36	20.49	20.53	20.44	20.59	20.30

5.3 Experiments with Watermark Attacks

In watermarking applications, an attack is defined as any operation that may hinder the detection or transmission of information included in a watermark. In considering the attacks on watermarks, the robustness feature of an algorithm becomes very important. In this regard, we classify a watermarking method as robust if the watermark data embedded by that algorithm in an image or any other data, cannot be damaged or removed without destroying or damaging the data itself. Therefore, an attack is successful if it can eliminate watermark without damaging the image itself.

The watermark attacks can be classified as removal attacks, geometric attacks, cryptographic attacks and protocol attacks. Removal attacks try to destroy the watermark data by applying some image improvement or enhancement algorithm. The idea comes from the fact that the watermark is added to the host data in the same way as noise is added. Therefore, any noise removing algorithm, such as smoothing or median filtering can remove watermark. Lossy compression algorithms also have the same impact as noise removal methods. One of the removal attacks is averaging. This attack is applicable if a large number of copies of the same image with different watermarks are available. In this case, averaging them will remove the watermark. Here the assumption is that the watermark has zero mean, which is a safe an reasonable assumption. Smoothing with a Gaussian filter and applying mean or median filters will also remove the watermark.

The geometric attacks do not remove watermarks but make detection impossible. The main geometric attacks are based on applying spatial transforms which affect the synchronization of the detector with the embedded watermark data. Shearing, rotation, row or column elimination, and scaling are among such attacks. Figure 5.2 shows an image of grid lines exposed to geometric distortion. This kind of distortion is not noticeable by the human visual system, and it makes detecting watermark quite difficult.

Cryptographic attacks are based on brute force methods to find the secret key. Short keys in applying watermark could be a good opportunity for cryptographic attackers to detect and extract watermark. Protocol attacks try to apply the inverse of the watermark. These type of attacks are applicable if the watermark algorithm is invertible. In this thesis we are considering only the removal and geometric attacks because our main interest is comparing the watermarking algorithms based on their robustness. Table 5.2 shows the attacks we have considered for evaluating the performance of the watermarking methods.

Some of the attacks given in Table 5.2 can be described as the removal attacks. These

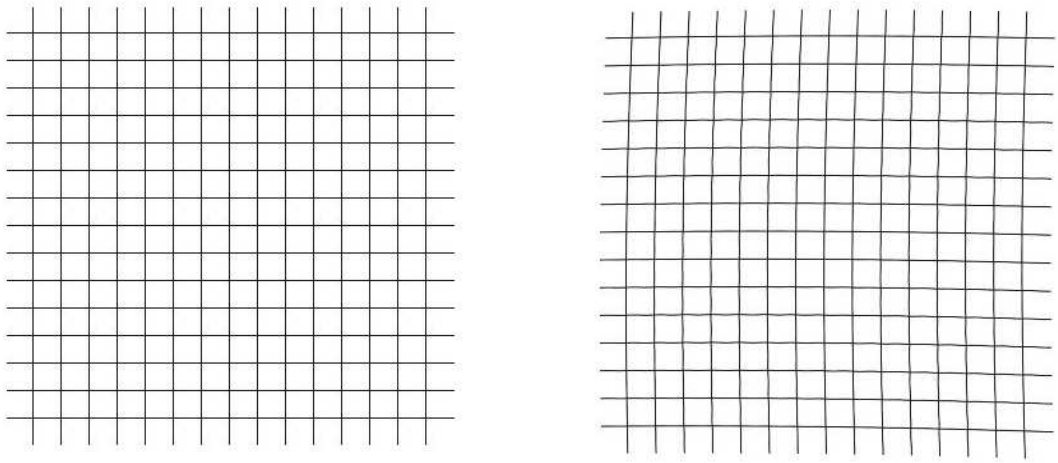

Figure 5.2: Geometric distortion of a grid.

attacks are based on modifying the data content of the image. In our selected list of attacks, median filtering, lossy JPEG and EZW compression, sharpening and Gaussian filtering fall into this group.

5.4 Evaluation of Robustness Against Attacks

In this section, we present our experimental results about the evaluation of robustness against the discussed attacks.

5.4.1 Experimental Setup

In this thesis, we focused on invisible watermark data. Watermark data size is 1000. This means that we have a string of coefficinets with length 1000. We used the discussed equations in Section 5.1 to measure the quality of the watermarked image with the original image. The tested images have different sizes such as 200×150, 200×200, 200×300, and 256×256. In our implementations, we have used the Matlab tools. The Matlab tools provide an environment for numerical computing and also for programming.

We performed two groups of experiments. These experiments are based on single bit and multiple bit watermarking. In the first group of experiments, we fixed the number of modified coefficients of DCT or DWT transforms to 1000. This is applied to the algorithms which are based on single bit watermarking methods. In the group of eight watermarking methods described and used in this research, Cox, Tsuan, Kim, Wang, Xia methods fall into this group. The algorithms proposed by Dugad, Koch, and Xie are multiple bit algorithms. For this group we have restricted the watermark length to

Table 5.2: List of attacks applied to the test images.

Attack	**Options**	
JPEG Compression	Quality: 10, 15, 20, 25, 30, 35, 40, 50, 60, 70, 80, 90	
EZW Compression	Using Haar wavelet bits-per-pixel: 0.01, 0.02, 0.05, 0.10, 0.15, 0.20, 0.25, 0.50, 0.75, 1.00, 1.25, 1.50	
Median Filtering	Filter sizes: 2×2 to 8×8	
Cropping	% of original image: 1, 2, 5, 10, 15, 20, 25, 50, 75	
Rotate + Scale	Angles: -2, -1, -0.75, -0.5, -0.25, 0.25, 0.5, 0.75, 1, 2, 5, 10, 15, 30, 45, 90	
Row-Column Removal	# of Rows removed	# of Columns removed
	1	1
	1	5
	5	1
	17	5
	5	17
Up-Down Scaling	Scale factors: 0.5, 0.75, 0.9, 1.1, 1.5, 2.0	
Shearing	x	y
	0.0	1.0
	0.0	5.0
	1.0	0.0
	1.0	1.0
	5.0	0.0
	5.0	5.0
Sharpening	Filter size 3×3	
Gaussian Filtering	Filter size 3×3	

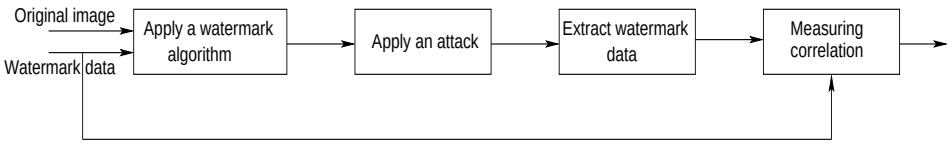

Figure 5.3: Block diagram for watermark robustness experiments.

64 bits. The block diagram given in Figure 5.3 indicates how the experiments have been applied to the test images.

First of all, each picture is watermarked using the methods described in the previous chapter, then an attack is applied. Next we try to extract the watermark and compute the amount of damage done to the watermark. The similarity between damaged watermark extracted from the image after the attack, and the original watermark is measured using their correlation. These correlation values have been averaged for each watermarking

algorithm and plotted against the parameters of each attack separately. We have used the following formula to find the similarity of two vectors

$$1 - \frac{1}{n \times \max(W_E, W_x)} \sum_{i=1}^{n} |W_E(i) - W_x(i)|. \tag{5.3}$$

where W_x is the embedded watermark, W_E is the extracted watermark, and n is the length of the embedded watermark.

We provide the results obtained by applying the attacks to the experimental images in the follwoing section.

5.4.2 Experimental Results

Figures 5.4 and 5.5 show the result of JPEG compression attack with single bit algorithms and multiple bit algorithms using 64-bit watermark respectively.
JPEG compression is a lossy compression. In the JPEG compression procedure, first a color space transform is performed from RGB to YCbCr. Then, Cb and Cr components are down-sampled by removing every other row and column. This corresponds to 50% compression. Thereafter, a DCT transform is applied and the DCT coefficients are quantized using a pre-defined quantization table. The JPEG compression rate depends on the quantization level used. Hence, higher compression ratio corresponds to lower image fidelity. It should be noted that quantization tends to eliminate high frequency data which corresponds to image details, noise, and embedded watermark.

Embedded Zero tree Wavelet (EZW) is a lossy compression based on organizing wavelet coefficients in a spatial tree. The branches with values less than the threshold are replaced with a zero tree or null tree, while for transmission, only a zero tree symbol is transmitted for the whole branch. The threshold value determines the compression rate and quality of the image. Similar to JPEG transform, EZW tends to destroy the high frequency content of the image and hence damage the watermark. Figures 5.6 and 5.7 show the result of EZW compression for single bit and multiple bit algorithms, respectively.

A median filter is normally used for eliminating noise in the images. Pepper and Salt noise can be easily removed by median filter. Median filter replaces the pixel under the center of the window with the middle value of the sorted pixel values falling into the window. Median filter also tends to destroy high frequency content and therefore, it damages the embedded watermark. The results obtained by applying median filtering are depicted in Figures 5.8 and 5.9.

Cropping attack refers to cutting part of the image. Generally speaking, cropping is used for the removal of the outer parts of an image to improve framing, emphasize subject matter or change aspect ratio. However, for watermark attack we have considered cropping the coefficients of DCT or DWT transform. Cropping is done after putting the coefficients in their order of importance by zig-zag scanning the coefficient matrix. This ordering of the coefficients puts the low frequency coefficients at the beginning of the list and hence tends to eliminate high frequency content of the image. Figures 5.10 and

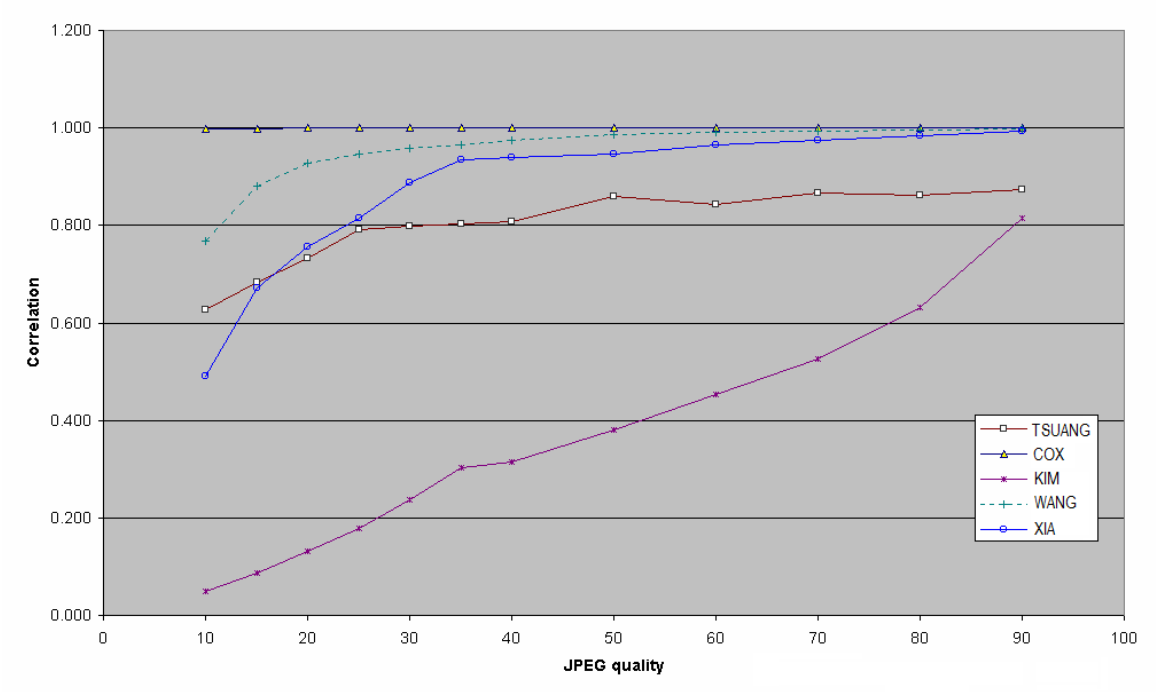

Figure 5.4: JPEG compression attack on single bit watermarking algorithms.

5.11 depict the results of cropping attacks on single bit and multiple bit algorithms, respectively.

Geometric attacks consider the image as a geometric object such as a rectangle and apply some simple geometric transforms which are not perceptible by the viewer. Among these transforms, rotation is more famous because it does not destroy the visual content of the image but due to rotation, some pixels move to new positions and the embedded watermark in that pixel is also moved to the new place. This displacement damages the embedded watermark. The amount of the damage introduced in this way depends on the image size, rotation angle and obviously on the method used for watermarking. The results obtained are depicted in Figures 5.12 and 5.13.

Watermark data is usually embedded on the neighboring pixels. Therefore, removing a row or column can damage the watermark. However, in cluttered natural images, removing a row or a column is not perceptible by the viewer. If the number of rows or columns removed from the image is large, they are replaced by their immediate neighbors. This replacement has the impact of eliminating image size change due to the removal of some rows or columns. The results of row and column removal attack on different watermark algorithms are shown in Figures 5.14 and 5.15.

Scaling image in a small rate has a similar effect as row/column removal (in case of down scaling). When up-scaling, in the simplest method the neighboring row/column is repeated and it is also possible to use interpolation for up-scaling the image. The

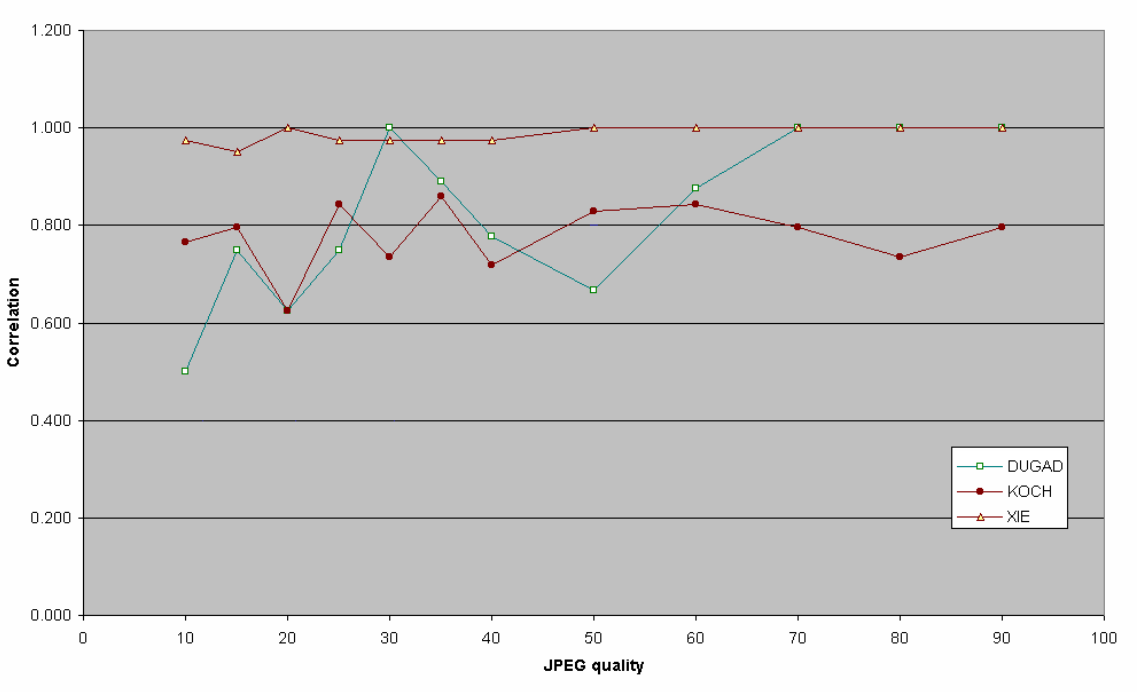

Figure 5.5: JPEG compression attack on multiple bit watermarking algorithms.

up/down scaling attacks used here are based on interpolating the missing values. The experimental results of applying the up-down scaling attacks on single bit and multiple bit watermarking algorithms are depicted in Figures 5.16 and 5.17.

Shearing attacks fall into the group of geometric attacks where some simple geometric transform is used to displace some of the pixels and hence, destroy the synchronization of the embedded watermark. Shearing is a affine transform preserving the image size and rotation. We have applied these attacks and the results are shown in Figures 5.18 and 5.19.

In general, Table 5.3 summarizes the results of applying different attacks to the watermarked images. The results given in this table are obtained by averaging the results of each method for a given attack over all parameters and all test images.

The results given in Table 5.3 indicate that Cox's algorithm has a better performance against DCT based JPEG algorithm. This can be related to the fact that JPEG uses the same transform (DCT) as used by Cox algorithm. One conclusion from this experiment is that when the watermarking and the attack are using the same transform or are in the same domain, for embedding the watermark we may choose the coefficients that are less affected by the attack and are less distorted. Cox method uses mid-frequency components to store the watermark, so attacks such as JPEG compression, median and Gaussian filtering which are low-pass filters do not have a major effect on it.

EZW lossy compression distortion rate is more compared to JPEG compression. This

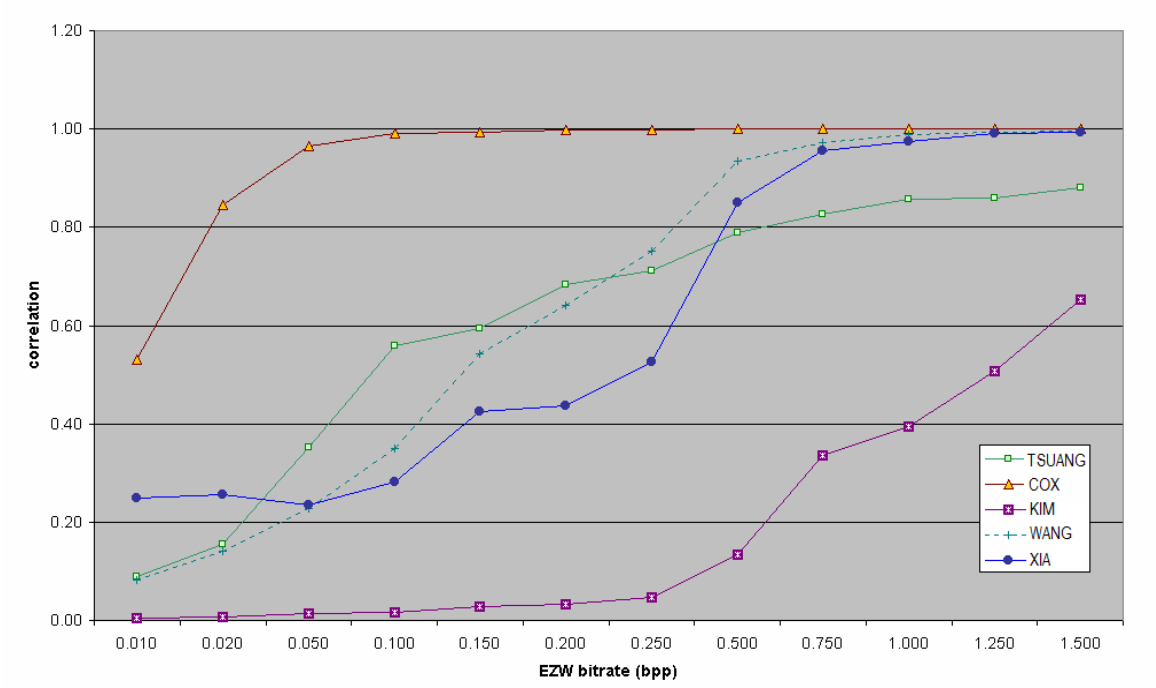

Figure 5.6: EZW compression attack on single bit watermarking algorithms.

Table 5.3: Test results in terms of correlation values for each watermarking method.

Attack	DUGAD	KOCH	XIE	TSUAN	COX	KIM	WANG	XIA
JPEG Compression	0.79	0.794	0.976	0.82	**0.99**	0.44	0.92	0.83
EZW Compression	0.54	0.49	0.69	0.60	**0.94**	0.28	0.61	0.55
Median Filter	0.24	0.23	0.72	0.58	**0.975**	0.18	0.48	0.40
Cropping	0.08	0.44	0.20	0.15	0.22	0.15	0.19	**0.61**
Rotation	0.22	0.38	0.24	0.21	**0.44**	0.06	0.11	0.30
Row, Column Removal	0.91	0.51	0.91	0.75	**1.00**	0.36	0.78	0.73
Scaling	0.81	0.49	0.93	0.77	**0.99**	0.52	0.79	0.74
Shearing	0.19	0.39	0.41	0.39	**0.59**	0.05	0.175	0.31
Sharpening 3×3	0.11	0.06	0.80	0.88	**0.98**	0.77	0.45	0.85
Gaussian filtering 3×3	**1.00**	0.47	**1.00**	0.88	0.99	0.29	0.77	0.71

can be related to the fact that EZW is a wavelet based method, which can achieve higher compression rates through its hierarchical coding. Median filtering attack uses a square window to find the median value. Normally the median filter can eliminate isolated noise. However, when the window size is even and hence the middle of the window is not the middle value in the list of pixel value, more distortion is caused. This fact is clear from Figure 5.8.

Geometric attacks as is clear form Figures 5.10, 5.11, 5.12, and 5.13 have the most drastic effect on the embedded watermark. Also, using the mid-frequency components

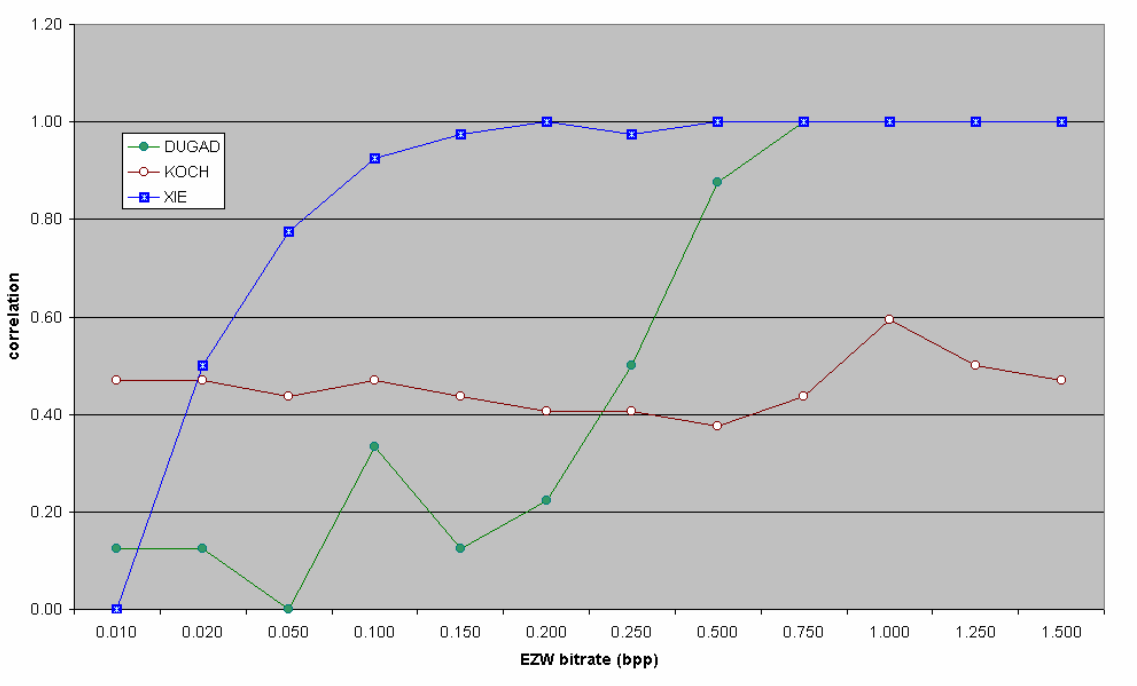

Figure 5.7: EZW compression attack on multiple bit watermarking algorithms.

make the watermark invisible. This gives it the robustness expected from a good watermarking method. For large filter sizes such as 15, the watermark is not completely destroyed and its correlation value remains at about 0.534. For other methods, even DWT based methods, achieving a correlation value this high, is not easy. Another algorithm which is comparable in performance to Cox algorithm is Xie's algorithm. However, Xie's algorithm modifies the wavelet coefficients using a median filter with a 1×3 sliding window. But despite this, its performance and robustness is very close to Cox's algorithm. Cox's algorithm has the advantage of preserving the original image at the watermark detector. This makes it possible for the detector to subtract the retrieved image and the original image and use the result as a metric to evaluate how good the watermark has been embedded. This helps the algorithm to avoid blind comparison and in fact because of this feature it falls into non-blind group of algorithms.

In blind algorithms, image is assumed to be noise and watermark constitutes the signal. The signal to noise ratio is quite low and the methods depend on correlation values for detection. As shown in Table 5.1 PSNR and MSE values for all algorithms are relatively high. This means that for better and more effective detection of the watermark we can increase the strength of the watermark. It should be noticed also that the rate of distortion in Cox algorithm is relatively high. The common specification of this method is that it is DCT based. All DWT based algorithms use a masking technique which in not available in DCT based methods. This is the main reason behind experimenting such a high rate of distortion in the DCT based methods compared to DWT based methods.

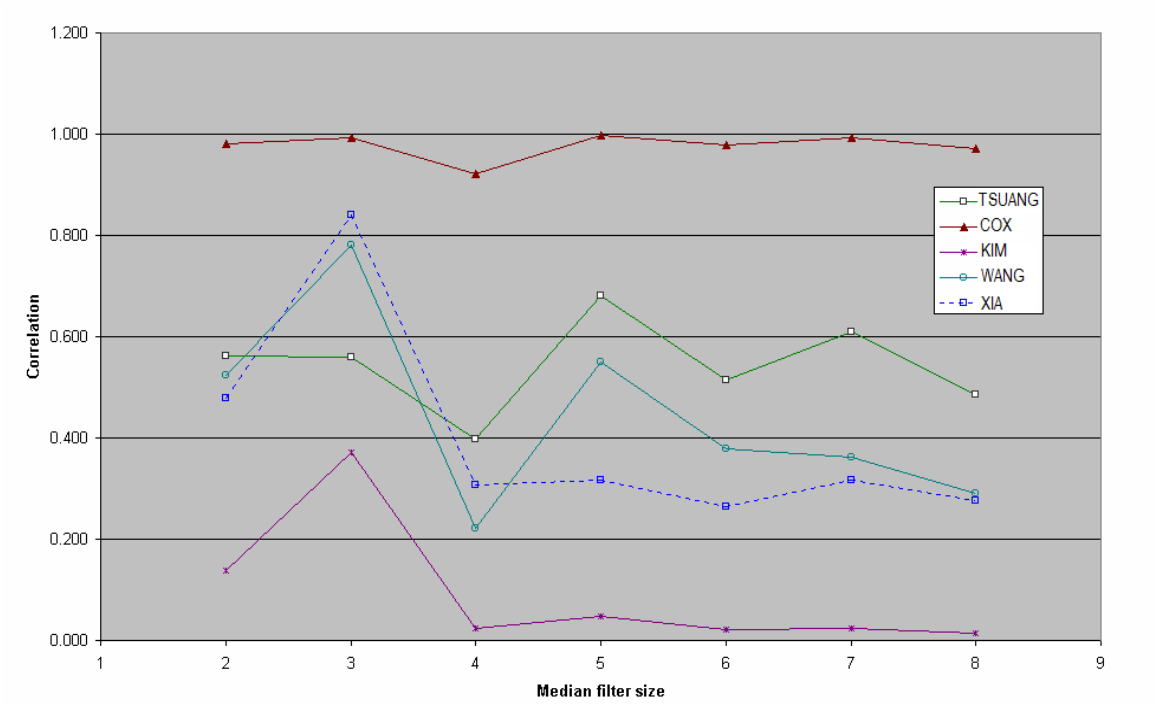

Figure 5.8: Median filtering attack on single bit watermarking algorithms.

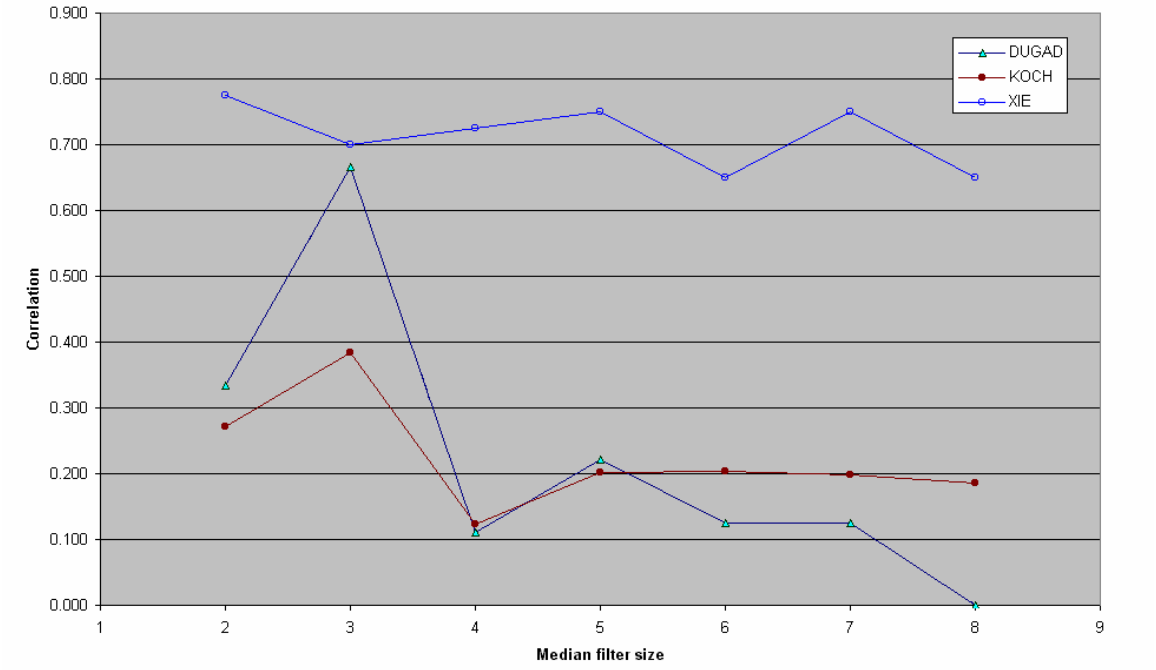

Figure 5.9: Median filtering attack on multiple bit watermarking algorithms.

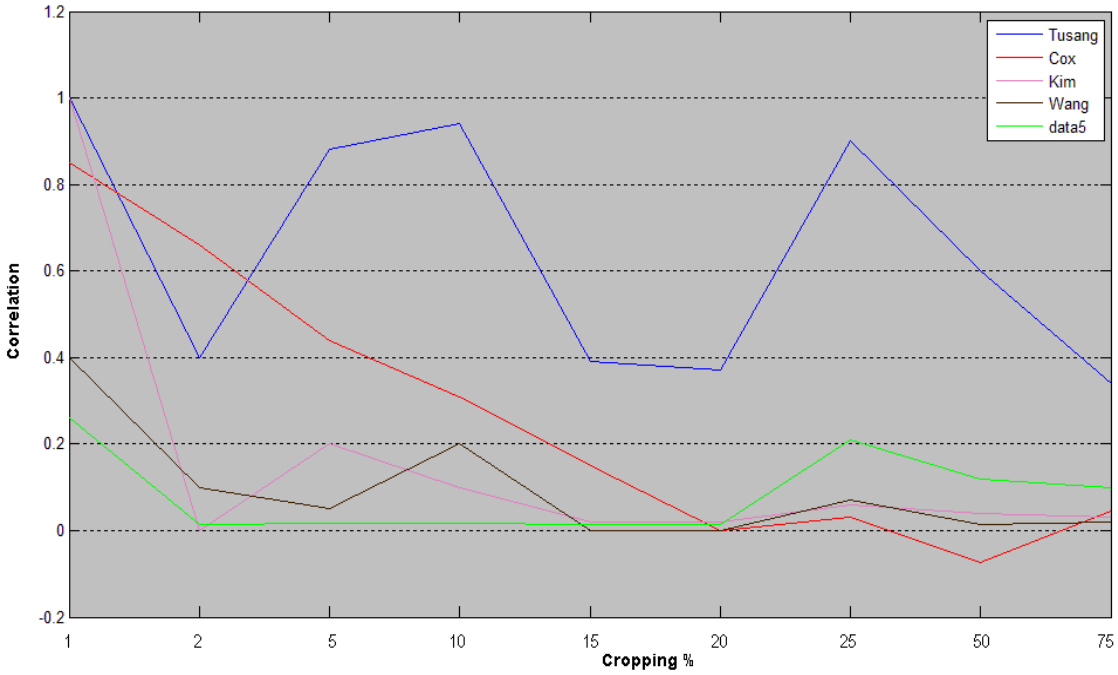

Figure 5.10: Cropping attack on single bit watermarking algorithms.

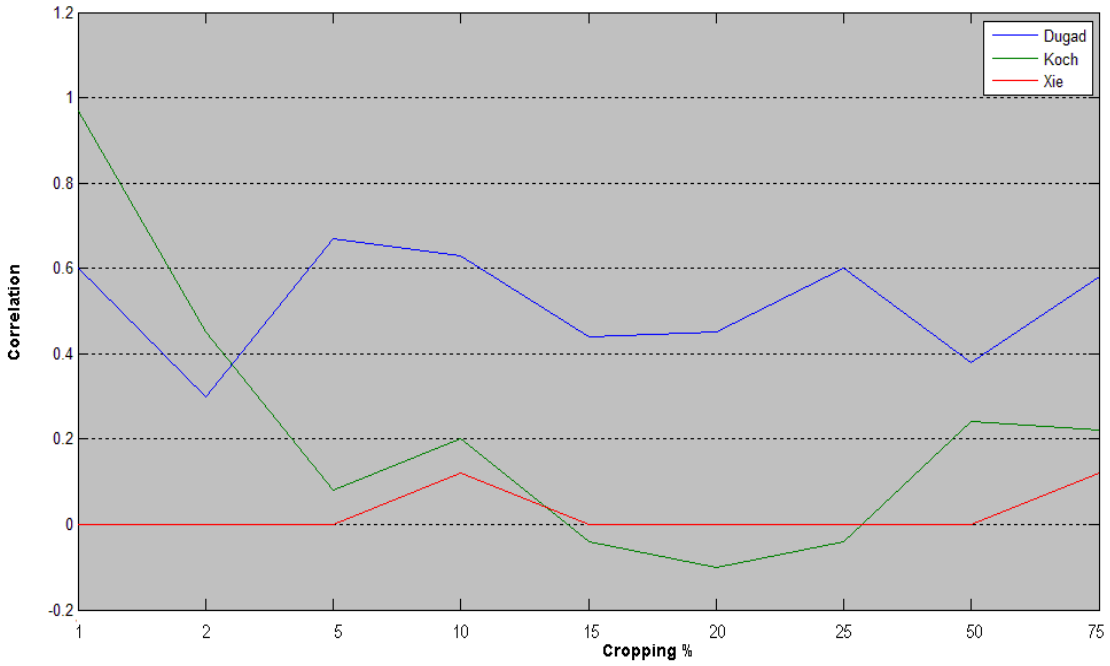

Figure 5.11: Cropping attack on multiple bit watermarking algorithms.

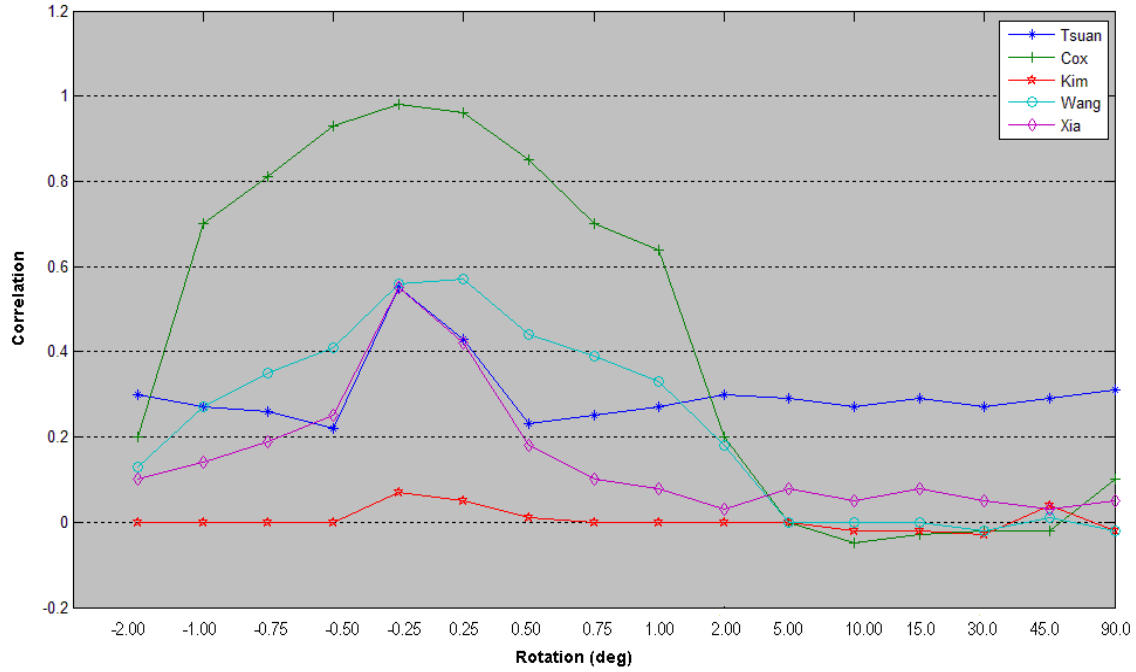

Figure 5.12: Rotation attack on single bit watermarking algorithms.

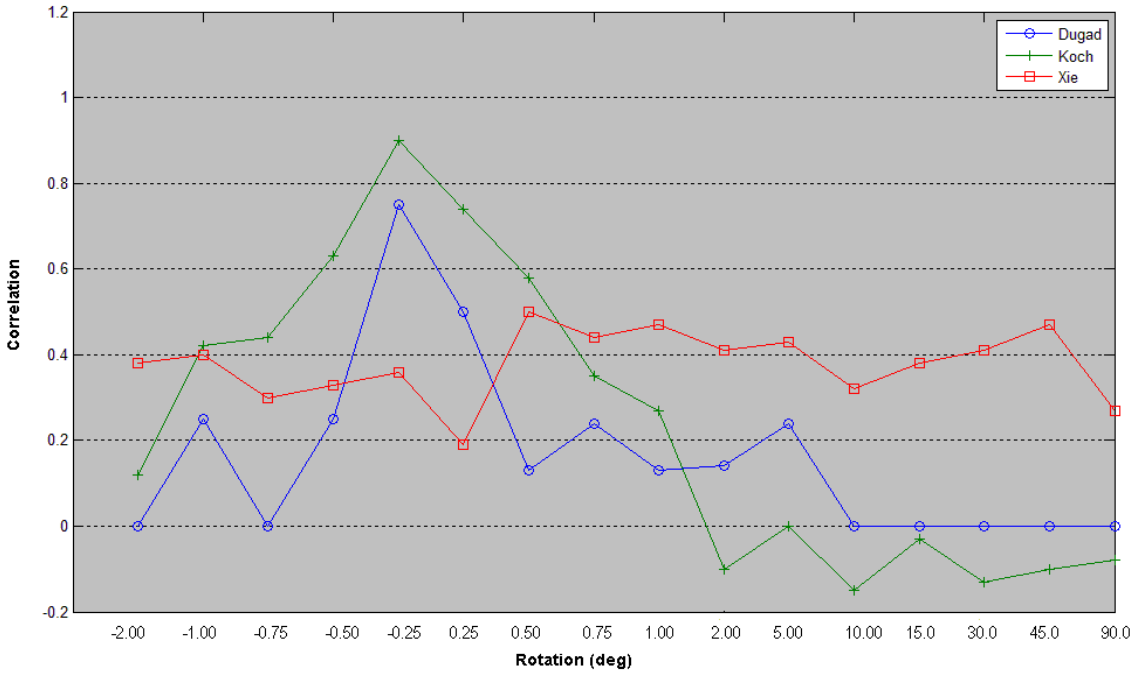

Figure 5.13: Rotation attack on multiple bit watermarking algorithms.

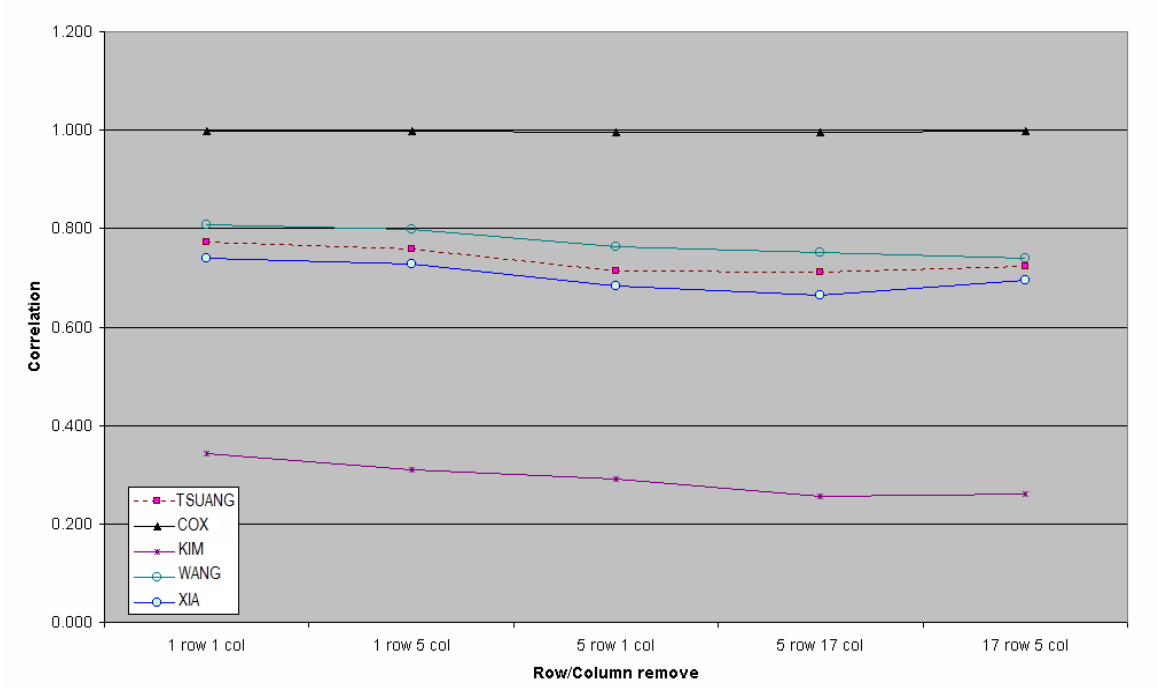

Figure 5.14: Row and column removal attack on single bit watermarking algorithms.

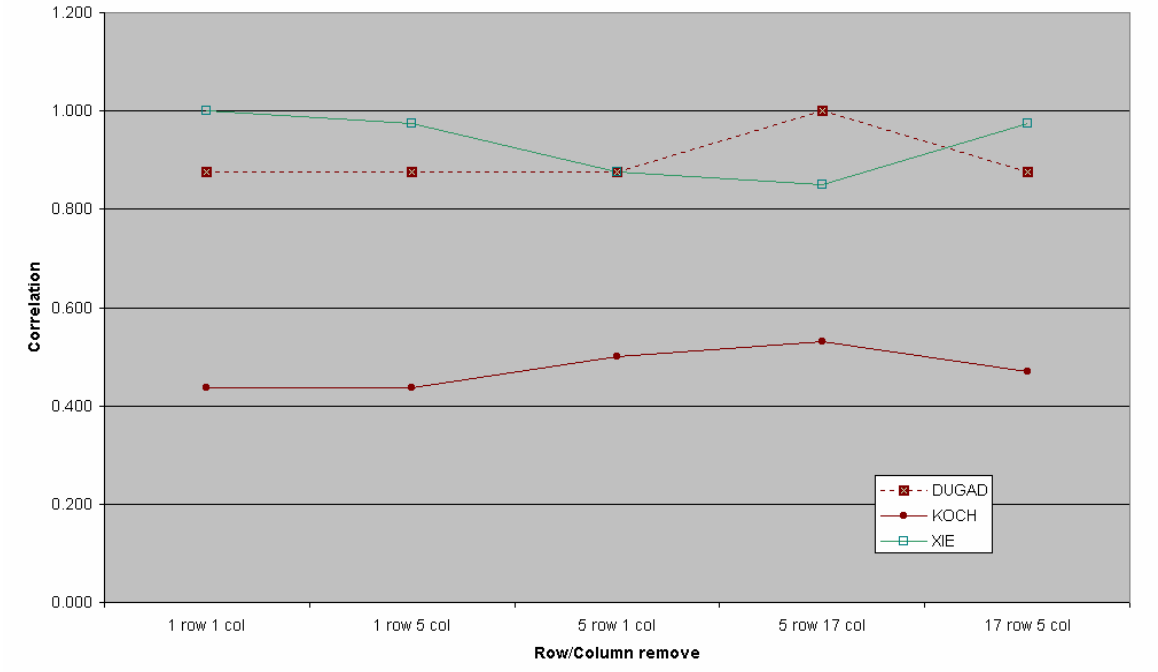

Figure 5.15: Row and column removal attack on multiple bit watermarking algorithms.

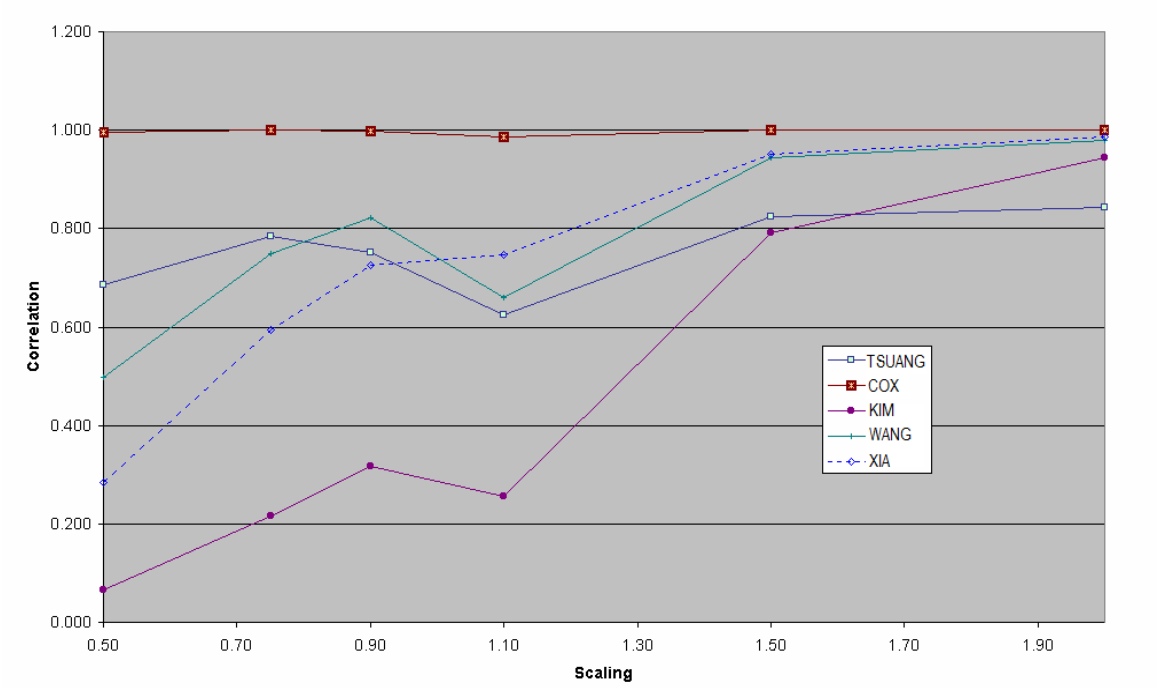

Figure 5.16: Up-down scaling attack on single bit watermarking algorithms.

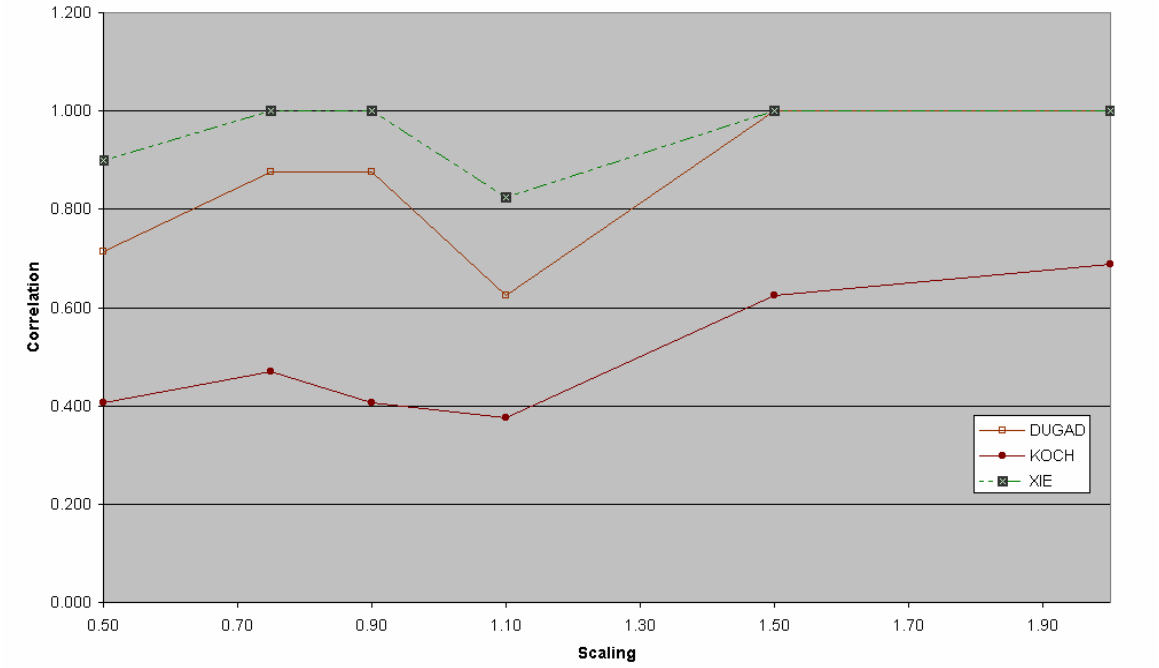

Figure 5.17: Up-down scaling attack on multiple bit watermarking algorithms.

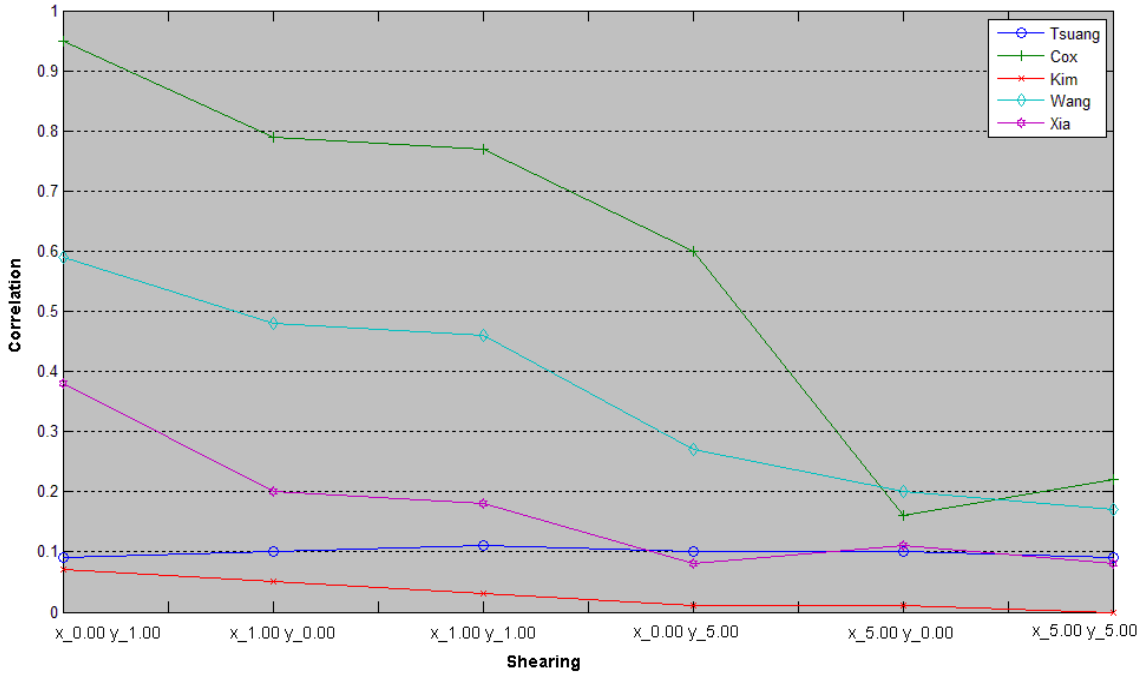

Figure 5.18: Shearing attack on single bit watermarking algorithms.

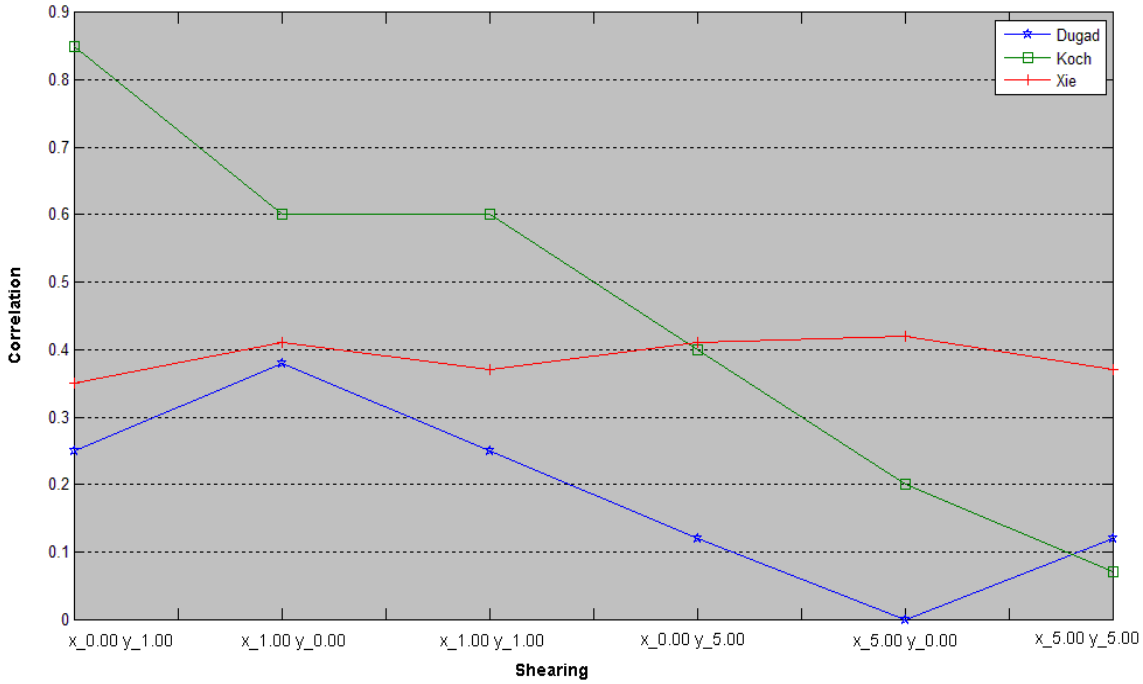

Figure 5.19: Shearing attack on multiple bit watermarking algorithms.

6 Conclusion and Future Work

Watermarking is the process of embedding predefined data into images in a way that the degradation of quality is minimized and remain in an imperceptible level. Many digital watermarking algorithms have been proposed in special and transform domains. The techniques in the spatial domain still have relative low-bit capacity and are not resistant enough to lossy image compression. On the other hand, frequency domain-based techniques can embed more bits for watermark and are more robust to attack. In this thesis, different transform watermark algorithms based on robustness have been evaluated. Specially, we compared the DCT and DWT watermarking algorithms with each other.

This chapter in organized as follows. In Section 6.1, we present a summary of the thesis followed by Section 6.2 where we explain some future work.

6.1 Summary

The robustness of transform domain watermarking algorithms on digital images have been evaluated using different attacks in this thesis. We have considered eight watermarking algorithms which are based on both DCT and DWT transforms. No spatial based watermarking has been examined because they are very sensitive to lossy compression and other removal attacks. The robustness of the watermarking methods has been quantitatively measured by comparing the extracted watermark with the original watermark. This helped us to measure the amount of damage done by each attack.

In two major cases, the watermark attacks destroy embedded watermark data. First when the watermark data is embedded in high frequency components of the images. High frequency content of an image behaves like added noise and therefore noise removal algorithms such as smoothing, and median filtering destroys it. Lossy compression algorithms also try to reduce the image size by removing the small details which correspond to high frequency content of the image. Therefore, watermarking methods such as Cox's algorithm which embed the watermark in mid frequency contents of the image are more robust against attacks. Secondly, when the watermarking algorithm and the attack are based on the same transform, the destructing effect of the attack is minimized. This is the case in Cox's algorithm when JPEG lossy compression attack is applied. Cox's algorithm and JPEG are both based on DCT transform. This study also can be used to adjust the strength of the watermark signal with respect to the image signal. The minimum desired correlation value can be used to adjust the power of the embedded watermark signal in a way that it is not affected by major attacks.

6.2 Future Work

This study described here measured the robustness of the available transform domain watermarking algorithms against major attacks. These attacks are aimed to destroy the watermark data in the image. As mentioned in Chapter **??**, there are other attacks such as brutal force attacks which try to detect and extract watermark data from the image. Our study can be extended to include those attacks and measure the robustness of the algorithms against them.

The robustness of a watermarking method depends on where (at what frequency contents) the watermark is embedded. As different images have different frequency contents, the robustness will also be dependent on the type of the image. In this study we have used a group of test images for measuring the robustness of each algorithm but we have not classified the images. An extension of our study therefore is classifying the images and applying the attacks on each group separately. In addition, applicability of the methods to other multimedia data such as video and audio can be verified and examined.

Another reseach work is the hardware implementation of watermarking algorithms. This is because a hardware implementation consumes less area and less power compared to a software implementation. Reconfigureable architectures such as Molen reconfigurable architecture [16, 33] can be used for hardware implementation. This is due to the fact that there are different watermark algorithms and each of them has its own requirements. In addition, transform domain watermark algorithms can be implemented using different DWT techniques that each of them uses various filter bank lengths and various transform levels.

Bibliography

[1] Y. Andreopoulos, K. Masselos, P. Schelkens, G. Lafruit, and J. Cornelis, *Cache Misses and Energy Dissipation Results for JPEG-2000 Filtering*, Proc. 14th IEEE Int. Conf. on Digital Signal Processing, 2002, pp. 201–209.

[2] Y. Andreopoulos, P. Schelkens, and J. Cornelis, *Analysis of Wavelet Transform Implementations for Image and Texture Coding Applications in Programmable Platforms*, Proc. IEEE Signal Processing Systems, 2001, pp. 273–284.

[3] Y. Andreopoulos, P. Schelkens, G. Lafruit, K. Masselos, and J. Cornelis, *High-Level Cache Modeling for 2-D Discrete Wavelet Transform Implementations*, Journal of VLSI Signal Processing **34** (2003), 209–226.

[4] Y. Andreopoulos, N. D. Zervas, G. Lafruit, P. Schelkens, T. Stouraitis, C. E. Goutis, and J. Cornelis, *A Local Wavelet Transform Implementation Versus an Optimal Row-Column Algorithm for the 2D Multilevel Decomposition*, Proc. IEEE Int. Conf. on Image Processing, vol. 3, 2001, pp. 330–333.

[5] C. Chrysafis and A. Ortega, *Line-Based, Reduced Memory, Wavelet Image Compression*, IEEE Trans. on Image Processing **9** (2000), no. 3, 378–389.

[6] A. Cohen, I. Daubechies, and J. C. F. Eauveau, *Biorthogonal Bases of Compactly Supported Wavelets*, Communications on Pure and Appl. Math. **45** (1992), no. 5, 485–560.

[7] I. Daubechies and W. Sweldens, *Factoring Wavelet Transforms into Lifting Steps*, Journal of Fourier Analysis and Applications **4** (1998), no. 3, 247–269.

[8] R. Dugad, K. Ratakonda, and N. Ahuja, *A New Wavelet-Based Scheme for Watermarking Images,*, Proc. IEEE Int. Conf. on Image Processing, 1998.

[9] F. Fang and S. Tan, *A Robust Digital Watermarking Technique with Improved Performance under JPEG Compression*, Proc. SPIE, Applications of Digital Image Processing, September 2006.

[10] M. Ferretti and D. Rizzo, *A Parallel Architecture for the 2D Discrete Wavelet Transform with Integer Lifting Scheme*, Journal of VLSI Signal Processing **28** (2001), 165–185.

[11] M. S. Hsieh, D. C. Tseng, and Y. H. Huang, *Hiding Digital Watermarks using Multiresolution Wavelet Transform*, IEEE Trans. on Industrial Electronics **48** (2006), no. 5, 875–882.

[12] J. Kilian I. J. Cox and T. Leighton Aand T. G. Shamoon, *Secure Spread Spectrum Watermarking For Multimedia*, IEEE Trans. on Image Processing **6** (1997), no. 12, 1673–1687.

[13] A. Kejariwal, S. Gupta, A. Nicolau, N. D. Dutt, and R. Gupta, *Energy Efficient Watermarking on Mobile Devices Using Proxy-Based Partitioning*, IEEE Trans. on Very Large Scale Integration (VLSI) Systems **14** (2006), no. 6, 625– 636.

[14] J. R. Kim and Y. S. Moon, *A Robust Wavelet-Based Digital Watermarking Using Level-Adaptive Thresholding*, Proc. IEEE Int. Conf. on Image Processing, October 1999, pp. 226–230.

[15] E. Koch and J. Zhao, *Towards Robust and Hidden Image Copyright Labeling*, Proc. IEEE Workshop on Nonlinear Signal and Image Processing, June 1995, pp. 452–455.

[16] G. K. Kuzmanov, G. N. Gaydadjiev, and S. Vassiliadis, *The Molen Media Processor: Design and Evaluation*, Proc. Int. Workshop on Application Specific Processors, September 2005, pp. 26–33.

[17] S. J. lee and S. H. Jung, *A Survey of WAtermarking Techniques Applied to Multimedia*, Proc. IEEE Int. Symp. on Industrial Electronics, June 2001.

[18] C. T. Li and H. Si, *Wavelet-Based Fragile Watermarking Scheme for Image Authentication*, Journal of Electronic Imaging **16** (2007), no. 1.

[19] C. Loeffler, A. Ligtenberg, and G. S. Moschytz, *Practical Fast 1-D DCT Algorithms With 11 Multiplications*, Proc. Int. Conf. on Acoustical and Speech and Signal Processing, May 1989, pp. 988–991.

[20] B. M. Macq and J. J. Quisquater, *Cryptology for Digital TV Broadcasting* , Proceedings of the IEEE **83** (1995), no. 1, 944–957.

[21] M. W. Marcellin, M. J. Gormish, A. Bilgin, and M. P. Boliek, *An Overview of JPEG 2000*, Proc. Data Compression Conf., March 2000.

[22] N. J. Mathai, D. Kundur, and A. Sheikholeslami, *Hardware Implementaion Perspectives of Digital Video Watermarking Algorithms*, IEEE Trans. on Signal Processing **51** (2003), no. 4, 925–938.

[23] S. R. M. Oliveira, M. A. Nascimento, and O. R. Zaiane, *Digital Watermarking: Status, Limitations and Prospects*, Tech. Report TR 02-01, University of Alberta, Department of Computer Science, January 2002.

[24] F. Petitcolas, *Photo database*, www.petitcolas.net/fabien/watermarking/imagedatabase/.

[25] M. Rabbani and R. Joshi, *An Overview of the JPEG2000 Still Image Compression Standard*, Signal Processing: Image Communication **17** (2002), no. 1, 3–48.

[26] A. Shahbahrami and B. Juurlink, *A Comparison of Two SIMD Implementations of the 2D Discrete Wavelet Transform*, Proc. 18th Annual Workshop on Circuits, Systems and Signal Processing (ProRISC2007), November 2007.

[27] A. Shahbahrami, B. Juurlink, and S. Vassiliadis, *Improving the Memory Behavior of Vertical Filtering in the Discrete Wavelet Transform*, Proc. 3rd ACM Int. Conf. on Computing Frontiers, May 2006, pp. 253–260.

[28] M. Steinebach, S. Zmudzinski, and F. Chen, *The Digital Watermarking Container: Secure and Efficient Embedding*, Proc. ACM Multimedia and Security Workshop, Sebtember 2004.

[29] L. D. Strycker, P. Termont, J. Vandewege, J. Haitsma, T. Kalker1, M. Maes, and G. Depovere, *Implementation of a Real-Time Digital Watermarking Process for Broadcast Monitoring on a TriMedia VLIW Processor*, IEE on Vision, Image and Signal Processing **147** (2000), no. 4, 371–376.

[30] J. Tao, A. Shahbahrami, B. Juurlink, R. Buchty, W. Karl, and S. Vassiliadis, *Optimizing Cache Performance of the Discrete Wavelet Transform Using a Visualization Tool*, Proc. 9th IEEE Int. Symp. on Multimedia, December 2007.

[31] M. A. Trenas, J. Lopez, E. L. Zapata, and F. Arguello, *A Memory System Supporting the Efficient SIMD Computation of the Two Dimensional DWT*, Proc. IEEE Int. Conf. on Acoustics Speech and Signal Processing, vol. 3, May 1998, pp. 1521–1524.

[32] R. G. van Schyndel, A. Z. Tirkel, and C. F. Osborne, *A Digital Watermark*, Proc. IEEE Int. Conf. on Image Processing, Sebtember 1994, pp. 86–90.

[33] S. Vassiliadis, S. Wong, G. N. Gaydadjiev, K.L.M. Bertels, G.K. Kuzmanov, and E. Moscu Panainte, *The Molen Polymorphic Processor*, IEEE Transactions on Computers (2004), 1363–1375.

[34] H. M. Wang, P. C. Su, and C. C. J. Kuo, *Wavelet-Based Digital Image Watermarking*, Optics Express **3** (1998), no. 12, 491.

[35] R. B. Wolfgang and E. J. Delp, *A Watermark for Digital Images*, Proc. IEEE Int. Conf. on Image Processing, 1996, pp. 219–222.

[36] X. G. Xia, C. G. Boncelet, and G. R. Arce, *Wavelet Transform Based Watermark for Digital Images*, Optics Express **3** (1998), no. 12, 497.

[37] L. Xie and G. R. Arce, *Joint Wavelet Compression and Authentication Watermarking*, Proc. Int. Conf. on image processing, October 1998, pp. 427–431.

[38] Y. Yusof and O. O. Khalifa, *Digital Watermarking For Digital Images Using Wavelet Transform*, Proc. IEEE Int. Conf. on Telecomunicatios, May 2007.