

**DEVELOPMENT AND ANALYSIS OF SPATIAL DOMAIN AND  
TRANSFORM DOMAIN WATERMARKING TECHNIQUE**

By

CHUA KAH KEONG

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements for the degree of

BACHELOR OF INFORMATION TECHNOLOGY (HONS)

COMPUTER ENGINEERING

Faculty of Information and Communication Technology

Department of Information Technology and Engineering

April 2011

UNIVERSITI TUNKU ABDUL RAHMAN

**REPORT STATUS DECLARATION FORM**

Title: DEVELOPMENT AND ANALYSIS OF  
SPATIAL DOMAIN AND TRANSFORM DOMAIN  
WATERMARKING TECHNIQUE

Academic Session: January 2011

I CHUA KAH KEONG  
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in

Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

\_\_\_\_\_  
(Author's signature)

\_\_\_\_\_  
(Supervisor's signature)

**Address:**

S-3, Wellesley Lunas Estate,

09600 Lunas,

Kedah.

\_\_\_\_\_  
Supervisor's name

**Date:** \_\_\_\_\_

**Date:** \_\_\_\_\_

## **DECLARATION OF ORIGINALITY**

I declare that this report entitled  
**“DEVELOPMENT AND ANALYSIS OF SPATIAL DOMAIN AND  
TRANSFORM DOMAIN WATERMARKING TECHNIQUE”**  
is my own work except as cited in the references.

The report has not been accepted for any degree and is not being submitted  
concurrently in candidature for any degree or other award.

Signature : \_\_\_\_\_

Name : \_\_\_\_\_

Date : \_\_\_\_\_

## **ACKNOWLEDGEMENT**

This dissertation will not have been made possible without the aid and guidance of numerous individuals who in one way or another contributed, furnish and extended their valuable assistance in the preparation as well as completion of this study.

First and foremost, my utmost gratitude to, Mr. Leong Chun Farn, my Final Year Project supervisor whose sincerity and encouragement I will never forget. Mr. Leong has been my inspiration as I hurdle all the obstacles in the completion of this project. I sincerely thank to my academic advisor, Dr. Aissa Boudjella, who rendered his advocacy and advice during the period of my project.

Last but not least, I wish to avail myself of this opportunity, express a sense of sincere benediction and love to my friends and my beloved family for their support, strength, help and everything.

## **ABSTRACT**

The main bourn of this Final Year Project is to develop and analyze different types of watermarking algorithms from spatial domain method and transform domain method.

Research and journal papers available currently are mainly focus in single type of watermarking method, thus readers have to spend more time in searching and reading papers of different watermarking method, indeed a time wasting yet less effective way. Under these circumstances, an effulgent idea of analyzing different watermarking techniques is proposed. This project will analyze 2 types of watermarking algorithms from spatial and transform domain, the algorithms chosen are Discrete Cosine Transform (DCT) and spatial domain.

Throughout the project, these 2 algorithms will be concisely studied and the pros and cons of each algorithm will be explained in detail. In a nutshell, this project will analyze the methods chosen in terms of robustness, imperceptibility, quality of embedded image, speed, security and complexity. Hence, a report which contains these 2 algorithms will be documented, alongside with depiction of comparison tables. The results from this project will greatly benefit researchers as it's useful in understanding range of watermarking techniques and comparison can be made easily, thus act as stepping stone for research purpose and future application of watermarking.

## TABLE OF CONTENT

<b>REPORT STATUS DECLARATION FORM.....</b>	<b>i</b>
<b>DECLARATION OF ORIGINALITY.....</b>	<b>ii</b>
<b>ACKNOWLEDGEMENT .....</b>	<b>iii</b>
<b>ABSTRACT .....</b>	<b>iv</b>
<b>TABLE OF CONTENT .....</b>	<b>v</b>
<b>LIST OF FIGURES.....</b>	<b>viii</b>
<b>LIST OF TABLES.....</b>	<b>ix</b>
<b>LIST OF ABBREVIATIONS .....</b>	<b>x</b>
<b>Chapter 1 INTRODUCTION.....</b>	<b>1</b>
1.1 Background .....	1
1.2 Motivation and Problem Statement.....	2
1.3 Project Scope and Objective .....	3
1.3.1 Objective .....	3
1.3.2 Project Scope .....	3
<b>Chapter 2 LITERATURE REVIEW.....</b>	<b>4</b>
2.1 Features of Watermark .....	4
2.2 Methods of Watermarking .....	5
2.3 Watermark Attributes .....	5
2.4 Classification of Watermark.....	6
2.5 Watermarking Applications .....	8
2.6 Attacks toward Watermark.....	9
2.7 Benchmark Tools for Watermarking Applications .....	10

<b>Chapter 3 METHODOLOGY.....</b>	<b>11</b>
3.1 DCT.....	11
3.1.1 Watermark Embedding Using DCT Method .....	11
3.1.2 Watermark Extraction Using DCT Method .....	16
3.1.3 DCT Watermark Embedding Flowchart .....	18
3.1.4 DCT Watermark Extraction Flowchart .....	19
3.2 DWT .....	20
3.2.1 Watermark Embedding Using DWT Method .....	20
3.2.2 Watermark Extraction Using DWT Method .....	21
3.2.3 DWT Watermark Embedding Flowchart .....	22
3.2.4 DWT Watermark Extraction Flowchart .....	23
3.3 Spatial Domain.....	24
3.3.1 Watermark Embedding Using Spatial Domain Method .....	24
3.3.2 Watermark Extraction Using Spatial Domain Method .....	24
3.3.3 Spatial Domain Watermark Embedding Flowchart .....	25
3.3.4 Spatial Domain Watermark Extraction Flowchart .....	26
3.4 Timeline .....	27
3.4.1 Phase 1: Literature Review Phase.....	27
3.4.2 Phase 2: Research Methodology Phase.....	27
3.4.3 Phase 3: Development Phase .....	27
3.4.4 Phase 4: Adjustment and Documentation Phase.....	28
3.4.5 Milestone.....	28
3.5 Development Tools .....	29

<b>Chapter 4 SIMULATIONS AND RESULTS .....</b>	<b>30</b>
4.1    Limitations .....	31
4.2    Graphical User Interface (GUI).....	32
4.3    Simulations.....	37
4.3.1    Simulation Set 1: Attack Free .....	37
4.3.2    Simulation Set 2: CKK Watermark.....	38
4.3.3    Simulation Set 3: UTAR Watermark .....	39
4.4    Simulation Result.....	40
4.4.1    Simulation Set 1 Result.....	40
4.4.2    Simulation Set 2 Results .....	43
4.4.3    Simulation Set 3 Result.....	70
4.5    Discussion .....	97
4.5.1    Simulation Set 1 .....	97
4.5.2    Simulation Set 2 & 3.....	98
<b>Chapter 5 CONCLUSION AND FUTURE WORK .....</b>	<b>100</b>
<b>REFERENCE .....</b>	<b>101</b>
<b>APPENDIX A: BIWEEKLY REPORT .....</b>	<b>A-1</b>
<b>APPENDIX B: MATLAB CODES .....</b>	<b>B-1</b>
DCT Embed.....	<b>B-Error! Bookmark not defined.</b>
DCT Extract .....	<b>B-Error! Bookmark not defined.</b>
Spatial Embed .....	<b>B-Error! Bookmark not defined.</b>
Spatial Extract .....	<b>B-Error! Bookmark not defined.</b>



## LIST OF FIGURES

Figure 2.1	Watermark Categorization	7
Figure 3.1	Permutation Mapping	12
Figure 3.2	Mapping of DCT coefficient	13
Figure 3.3	DCT Watermark Embed Flowchart	18
Figure 3.4	DCT Watermark Extract Flowchart	19
Figure 3.5	DWT Watermark Embed Flowchart	22
Figure 3.6	DWT Watermark Extract Flowchart	23
Figure 3.7	Spatial Domain Embed Flowchart	25
Figure 3.8	Spatial Domain ExtractFlowchart	26
Figure 4.1	Host and watermark images	30
Figure 4.2	Main menu for DCT and spatial domain watermarking	32
Figure 4.3	Embed menu for DCT and spatial domain watermarking	33
Figure 4.4	Extract menu for DCT and spatial domain watermarking	34
Figure 4.5	Attack menu for DCT and spatial domain watermarking	35
Figure 4.6	PSNR calculation menu for DCT and spatial domain watermarking	36
Figure 4.7	Watermarked images of size 512 x 512 pixels	40

## LIST OF TABLES

Table 2.1	Characteristic of Watermark Algorithms	6
Table 2.2	Types of Attack	9
Table 3.1	Milestone	28
Table 4.1	Attacks for CKK watermark	38
Table 4.2	Attacks for UTAR watermark	39
Table 4.3	SNR comparisons between DCT and spatial algorithm watermarked images	41
Table 4.4	PSNR comparisons between DCT and spatial algorithm watermarked images	41
Table 4.5	NCC comparisons between DCT and spatial algorithm watermarked images	42
Table 4.6	PSNR comparison for Lena and CKK watermarked images	43
Table 4.7	NCC comparison for Lena and CKK watermarked images	45
Table 4.8	PSNR comparison for Baboon and CKK watermarked image	52
Table 4.9	NCC comparison for Baboon and CKK watermarked image	54
Table 4.10	PSNR comparison for Pepper and CKK watermarked image	61
Table 4.11	NCC comparison for Pepper and CKK watermarked image	63
Table 4.12	PSNR comparison for Lena and UTAR watermarked images	70
Table 4.13	NCC comparison for Lena and UTAR watermarked image	72
Table 4.14	PSNR comparison for Baboon and UTAR watermarked images	79
Table 4.15	NCC comparison for Baboon and UTAR watermarked images	81
Table 4.16	PSNR comparison for Pepper and UTAR watermarked images	88
Table 4.17	NCC comparison for Pepper and UTAR watermarked images	90

## LIST OF ABBREVIATIONS

DCT	Discrete Cosine Transform
DWT	Discrete Wavelet Transform
DFT	Discrete Fourier Transform
RGB	colour map that maps image information into Red (R), Green (G) and Blue (B) channel.
HSI	Hue, Saturation, Intensity
CMY	Cyan, Magenta, Yellow
Y'UV	colour space used as part of colour image, Y' stands for luma component (brightness), U and V are chrominance (colour) components.
GUI	Graphical User Interface
SNR	Signal to Noise Ratio
PSNR	Peak Signal to Noise Ratio
NCC	Normalized Cross Correlation

## **Chapter 1 INTRODUCTION**

### **1.1 Background**

Nowadays, digital watermarking is having a great vogue all around the world, due to security and piracy issues. Rapid improvement of technology has break the bond limiting people to share digital documents such as images, videos, audios and texts, which is the prime mover of germination and duplication of someone's masterpiece without the owner's sanction. Hence, it is a great hindrance for us to conserve the owner and make sure the customers are out of harm's way at the same time.

Under these circumstances, watermarking has become the most desirable solution to embark upon this matter. Embedding watermark into products or digital contents is a copyright protection for both the owner and authorized user or customer, as the watermarks can be extracted whenever it is needed for clarification.[1] It is a must to do so as the absence of copyright protection prevail those illicit copies into the market, thus causing the havoc on ownership of the source.

Watermarking is the process of embedding information into a multimedia component, for example, an image. Meanwhile, for security or piracy detection purpose, the information embedded can be detected or extracted out from its host [2], without causing damage to its host. Essentially, digital watermark is a code that is embedded into an image, which acts as digital signature, thus providing ownership to the image. [3]

## 1.2 Motivation and Problem Statement

In this modern era of technology, watermarking plays an important role in preventing piracy. However, our knowledge about watermarking is not fully-fledged and still has rooms for improvement along with enhancement. Digital watermarking applications are rare yet not dexterous enough, even though some of the applications can be found as freeware through the World Wide Web, they're still not up to par.

Digital watermarking is a hot topic nowadays, there are many people study the different algorithms of watermarking and papers are being published. Currently, most of the papers or journals available are mainly focus in a specific algorithm. Thus, public are having a hard time when they are required to choose the better algorithm to be implemented, as they do not have sufficient information about different algorithms. Besides, they're incapable of providing the advantages and disadvantage of various watermarking methods in detail, unless they are willing to spend extra time collecting and reading papers of antithetic algorithms.

In order to solve this, we need to clear the path and provide concise explanation on different types of watermarking algorithms, so others can familiarize with different algorithms in shorter time and compare those algorithms easily.

## 1.3 Project Scope and Objective

### 1.3.1 Objective

The prime objective of this project is to develop and analyze different algorithms from spatial domain and transform domain watermarking method. The outcome of this project will benefit the researchers as it can precisely explain the advantages and disadvantages of each algorithm, thus quick comparison can be made without extra effort wasted in collecting the required information from scratch. After all the methods being analyzed, embed and extract process of each algorithm are further improvised to obtain a simple function which can be reused in the future.

### 1.3.2 Project Scope

The scope for the project can be sum up as below:

- (i) Implementation of 2 different algorithms from spatial domain and transform domain. For transform domain, DCT is chosen.
- (ii) Analyze the 2 methods chosen in terms of:
  - a. Robustness
  - b. Imperceptibility
  - c. Speed
  - d. Security
  - e. Complexity
- (iii) The host images for watermarking are Lena, Baboon and Peppers.
- (iv) Watermarks used are UTAR logo and CKK logo respectively.
- (v) The picture type supported is bitmap (.bmp) files.
- (vi) Stirmark will be used as benchmark tool for the algorithms, as it provides various types of attacks which are sufficient for this project, which are cropping, rotation, sharpening, Gaussian filtering, random bending, linear transformation, aspect ratio, line removal and color reduction.

## **Chapter 2 LITERATURE REVIEW**

Technology...is a queer thing. It brings you great gifts with one hand, and it stabs you in the back with another. A quote once cited by C.P. Snow, 1971. [4]

Amelioration of technology is like a house on fire, progressing at breakneck speed. Due to the rapid evolution in the internet traffic, alongside with its significance in content authentication and copyright protection for digital multimedia, digital image watermarking has drew increasing attention in the last few years [5] [6] by embedding information or data into the original image.

### **2.1 Features of Watermark**

An excellent watermark is having the capability to feature plenty of important characteristic. First of all, the watermark must be difficult to descry after it is embedded into the source, and the embedded watermark needs to be intuitively invisible so it will not deface the original image. Meanwhile, it ought to countervail malicious attacks and common distortions. Besides the features above, the watermark must have the ability to carry multitudinous bits of crucial information while able to coincide with other watermarks at the same time. [7]

If classical cryptography is applied [8], the encrypted signal will tend to become cluttered data, thus failed to pass the checkpoint on the network. Steganography [9], however, cater another layer of protection on the signal before embedding it into another media, such that the transmitted data is meaningful and harmless to others. Image steganographic can be separated into two: spatial-domain based method and frequency-domain based method. As for spatial-domain methods, watermarks are integrated into the depth of pixels of image directly. However, the adverse circumstance of spatial-domain watermarking is alteration on picture such as cropping, will eventually cast out the watermark embedded. On the other hand, frequency-domain methods will convert the original image into frequency domain before embedding the watermark in it.

## 2.2 Methods of Watermarking

There are several types of schemes for embedding the digital watermark, the most essential methods of watermarking are based on Discrete Cosine Transform (DCT) [10], Discrete Wavelet Transform (DWT) [11], Discrete Fourier Transform (DFT) [12], spatial-domain schemes [13], and vector quantization domain methods (VQ) [14]. DCT, DFT and DWT can be classified under transform-domain approach.

To sum things up, spatial-domain and transform-domain watermarking are construed as follows:

- ❖ Spatial Domain method:
  - (i) One or two subset of an image is opted at random for slight modification, such as flipping the low-order bit of each pixel. [15]
  - (ii) The downside is filtering or compression may render it useless.
  
- ❖ Transform method:
  - (i) Values of lower frequency levels are amended from their original image.
  - (ii) Higher frequencies are neglected due to data loss during compression.
  - (iii) The entire image is watermarked, thus cropping action will not remove the watermark.
  - (iv) Drawback of frequency domain is difficulties during verification as watermark is embedded at random in the image. [15]

## 2.3 Watermark Attributes

For an excellent watermark, it must fulfill several requirements and certain attributes [16], which can be explained as follows:

- Imperceptibility

Determine how close the watermarked image resembles the original image. A proper watermarked image will have the identical look with its original host image. The quality of embedded image can be calculated by

Peak Signal-to-Noise Ratio (PSNR).  $PSNR = 20 \log_{10} \left( \frac{MAX}{\sqrt{MSE}} \right)$



- **Robustness**  
Determine how good is a watermark can sustain on attacks applied on it. A good watermark can endure severe attacks, yet can still be extracted to prove one's ownership. Robustness of a watermark is determined through Normalized Cross Correlation (NCC). 
$$NCC = \frac{\sum_i \sum_j W(i,j) \hat{W}(i,j)}{\sum_i \sum_j [W(i,j)]^2}$$
- **Capacity**  
A good watermark must have maximized its data embedding payload. The ratio of watermark to host image determines how good the capacity of an algorithm is.
- **Security**  
For a proper watermarked image, any clues or hint of watermark will not be traced by others.

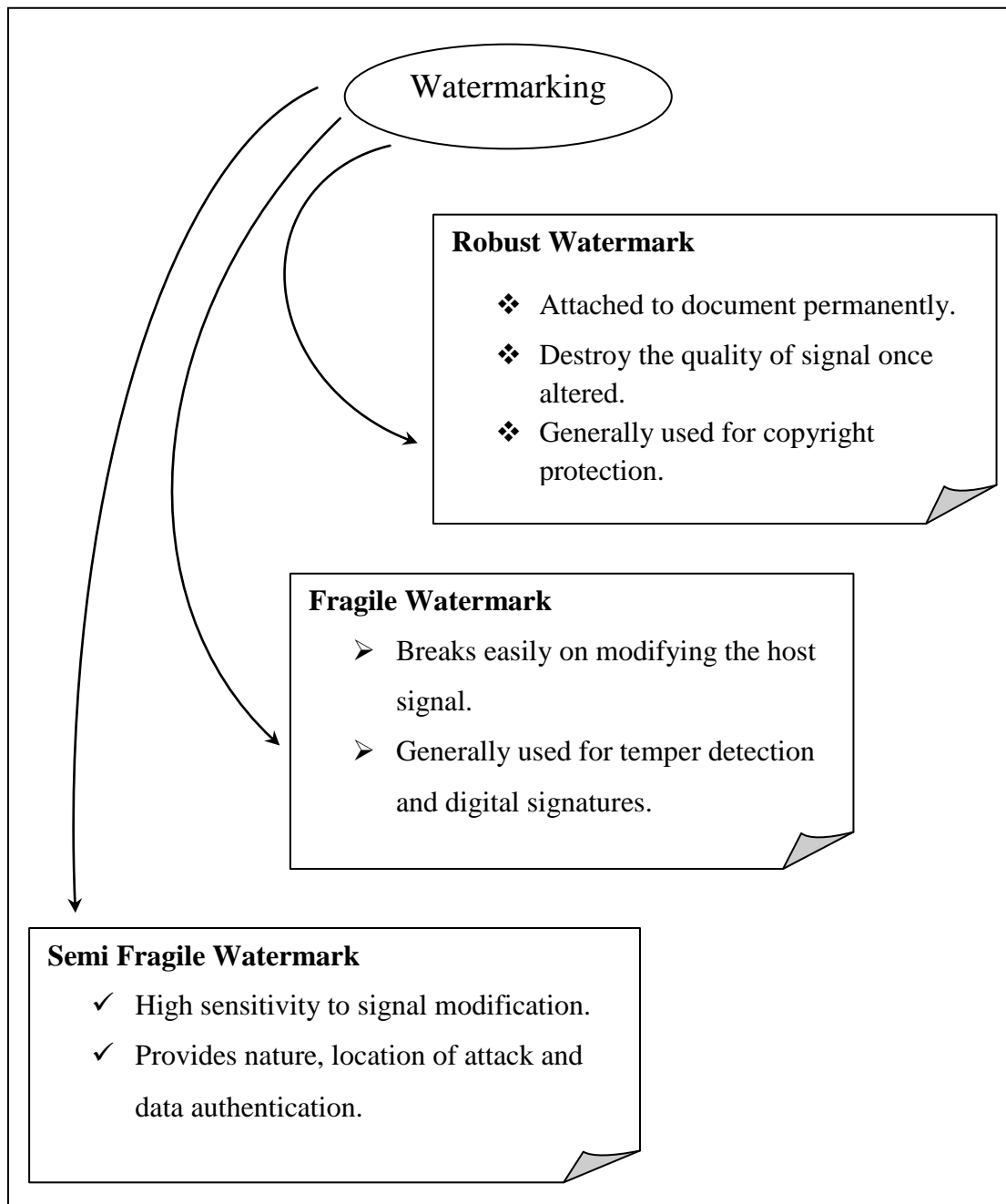
## 2.4 Classification of Watermark

Generally, watermarking algorithms can be divided into three, which are non blind, semi blind and blind [16]. Table 2.1 below concisely analyzes the characteristic of each algorithm:

Algorithm	Characteristic
Non Blind	Adopt the initial signal for watermark implementation process. Both secret keys and original image are used.
Semi Blind	Lateral information ( watermark bit sequence [6] ) alongside with the secret key are used in this algorithm.
Blind	None of the lateral data or the initial signal is used throughout the process. Only secret key is required. [6]

**Table 2.1 Characteristic of Watermark Algorithms**

Watermark can be classified into three categories, which are Robust Watermark, Fragile Watermark and Semi Fragile watermark [16]. below briefly explains all three categories of watermark:



**Figure 2.1 Watermark Categorization**

## 2.5 Watermarking Applications

Watermarking applications can be characterized in virtue of several properties. [17] [18]The priority of each property solely relies upon the prerequisite of system application:

- Embedding effectiveness
- Fidelity
- Data payload
- Blind / Informed detector
- False positive rate
- Robustness, security and cost

As for watermarking applications, it can be summarized as follows: [17]

- Copyright protection
  - ✓ To certify the ownership of certain content.
- Signatures
- Fingerprinting
  - ✓ By embedding watermarks, the initial buyer or owner can be distinguished, thus speed up the process of tracking illegitimate duplications.
- Broadcasting and publication monitoring
  - ✓ With the alleviation of automated systems, ownership of certain content is identified.
- Authentication
  - ✓ Crucial information is encrypted in order to prove that the content is genuine.
- Copy control
  - ✓ The watermark will control the action for user to manipulate or copy the content according to owner's will.
- Secret communication [18]
  - ✓ Signal is integrated within the transmission of secret data from one to another, without being noticed by anyone.

## 2.6 Attacks toward Watermark

Different types of digital watermarking have their own data encryption as well as level of security. Consequently, it does not entirely impenetrable to attacks done by users. Some prearranged or intentional attacks are shown in **Error! Reference source not found.** below: [19]

Types of Attack	Description
Active Attacks	Attempt to remove or render the watermark imponderable.
Passive Attacks	Watermark is unharmed, hacker attempt to verify the existence of watermark.
Collusion Attacks	Reconstruction of new image, using several copies of differently watermarked images.
Forgery Attacks	Instead of removing the watermark, hacker attempt to implement his own watermark into the image.
Distortive Attacks	Distortive transformation is applied to render the watermark undetectable. [20]

**Table 2.2 Types of Attack**

## 2.7 Benchmark Tools for Watermarking Applications

In order to evaluate the efficiency of certain digital watermarking application, benchmark tools are developed to carry out the task. There are quite a number of benchmark tools being used by programmers to standardize the watermarking application assessing process.

### 1) Stirmark [21]

- Designed to test robustness.
- Provided with an image, a number of remodeled images are generated to test whether the watermark still detectable.
- Attacks / Features available: cropping flip, rotation, rotation-scale, FMLR, sharpening, Gaussian filtering, Random bending, linear transformation, aspect ratio, scale changes, line removal, color reduction, JPEG compression. [22]

### 2) Checkmark

- Developed on Matlab under Microsoft and UNIX.
- Offers extra attacks which are not achievable in Stirmark.
- Attacks / Features available: wavelet compression, projective transformation, warping, copy, template removal, denoising, perceptual remodulation, non-linear line removal, collage. [22]

### 3) Optimark

- A tool developed to regulate inadequacy found in Stirmark 3.1.
- Attacks / Features available: GUI, detection performance evaluation, ROC curve, detection and embedding time evaluation, payload size evaluation. [22]

### 4) Certimark

- A benchmarking tool used for watermarking visual content, alongside with certification process for watermarking algorithms. [23]

## Chapter 3 METHODOLOGY

### 3.1 DCT

#### 3.1.1 Watermark Embedding Using DCT Method [24]

Assume  $X$  to be original image of size  $N_1 \times N_2$ , and the digital watermark,  $W$  to be a binary image of size  $M_1 \times M_2$ . Meanwhile, the marked pixels are valued as 1, and the rest are marked as 0. As only the middle-frequency range of the original image will be used, thus the resolution of the digital watermark  $W$  is assumed smaller than the host image,  $X$ . The original image  $X$  and the watermark  $W$  are represented as follows:

$$X = \{x(i, j), 0 \leq i \leq N_1, 0 \leq j \leq N_2\}$$

$$W = \{w(i, j), 0 \leq i \leq M_1, 0 \leq j \leq M_2\}$$

#### Step 1: Pseudorandom Permutation of The Watermark

Every watermark block will only be dispersed over its corresponding image block, but not the whole spatial image. Thus, in order to survive from picture cropping, a fast 2-D pseudorandom number traversing method is used to permute the watermark to disperse its ordinary spatial relationship, such that:

$$W_p = \text{Permute}(W)$$

$$W_p = \{w_p(i, j)\} \\ = \{w(i', j'), 0 \leq i, i' < M_1 \text{ and } 0 \leq j, j' < M_2\} \quad [25]$$

Where pixel  $(i', j')$  is permuted to pixel  $(i, j)$  in a pseudorandom order.

#### Step 2: Block-Based Image-Dependent Permutation of the Watermark

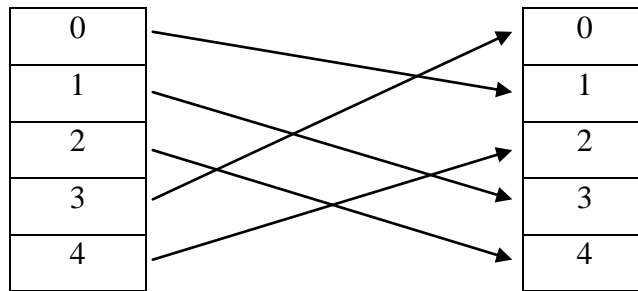
Each characteristic of the original image has to be considered in order to enhance and improve the perceptual invisibility. In this situation, every image block of size  $8 \times 8$ , its variances are calculated and sorted. In addition, for each watermark block of size  $(M_1 \times \frac{8}{N_1}) \times (M_2 \times \frac{8}{N_2})$ , the number of signed pixels are sorted as well.

Next will be the reshuffling of each watermark block into the spatial position according to the corresponding sorting order of the image block.

$$\begin{aligned} W_p = \{ & w_p(k \times (M_1 \times \frac{8}{N_1}) + i, l \times (M_2 \times \frac{8}{N_2}) + j), \\ & 0 \leq k < \frac{N_1}{8}, 0 \leq l < \frac{N_2}{8}, 0 \leq i < (M_1 \times \frac{8}{N_1}), \\ & 0 \leq j < (M_2 \times \frac{8}{N_2}) \} \end{aligned} \quad [26]$$

$$\begin{aligned} W_b = \{ & w_b(k \times (M_1 \times \frac{8}{N_1}) + i, l \times (M_2 \times \frac{8}{N_2}) + j) \\ = \{ & w_p(k' \times (M_1 \times \frac{8}{N_1}) + i, l' \times (M_2 \times \frac{8}{N_2}) + j), \\ & 0 \leq k, k' < \frac{N_1}{8}, 0 \leq l, l' < \frac{N_2}{8}, 0 \leq i < (M_1 \times \frac{8}{N_1}), \\ & 0 \leq j < (M_2 \times \frac{8}{N_2}) \} \end{aligned} \quad [27]$$

Permutation mapping of the watermark block



**Figure 3.1 Permutation Mapping**

### Step 3: Block Transformation of the Image

The original image X is divided into blocks of 8 x 8, and each block is DCT independently transformed.

$$Y = \text{FDCT}(X)$$

Where FDCT denotes the operation of forward DCT.

### Step 4: Choice of Middle-Frequency Coefficients

The middle-frequency coefficients are extracted from Y, due to human eyes has higher sensitivity towards noise in lower frequency compare to higher frequency. Watermark is embedded into middle-frequency range of image in order for the

watermark to survive in lossy data compression [24]. Out of 64 DCT coefficients, only  $(64 \times \frac{M_1 \times M_2}{N_1 \times N_2})$  coefficients are selected, then being mapped into a reduced image of block size  $(M_1 \times \frac{8}{N_1}) \times (M_2 \times \frac{8}{N_2})$ .

$$Y_r = \text{Reduce}(Y)$$

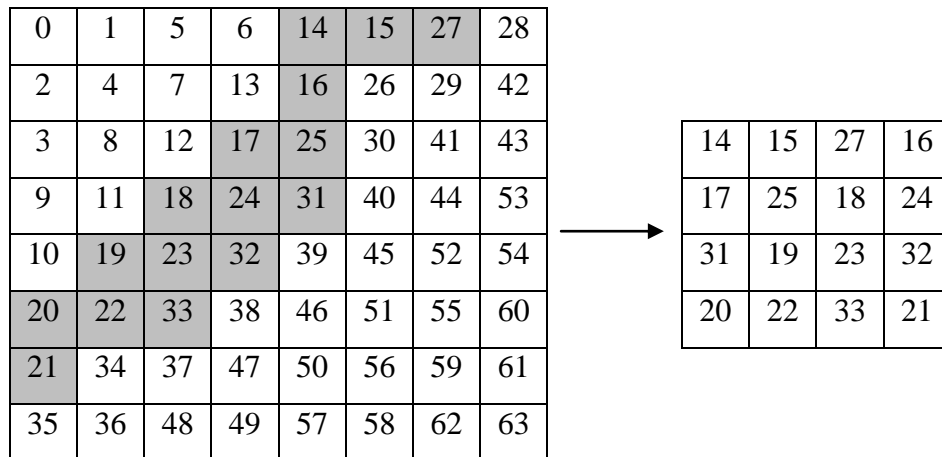
Where

$$Y = \{y(k \times 8 + i, l \times 8 + j), 0 \leq k < \frac{N_1}{8}, 0 \leq l < \frac{N_2}{8}, 0 \leq i < 8, 0 \leq j < 8\} \quad [28]$$

And

$$Y_r = \{y_r(k \times (M_1 \times \frac{8}{N_1}) + i', l \times (M_2 \times \frac{8}{N_2}) + j') \\ 0 \leq k < \frac{N_1}{8}, 0 \leq l < \frac{N_2}{8}, 0 \leq i' < (M_1 \times \frac{8}{N_1}) \\ 0 \leq j' < (M_2 \times \frac{8}{N_2})\} \quad [29]$$

**Error! Reference source not found.** below shows the DCT coefficients being picked and mapped into 4 x 4 block.



**Figure 3.2 Mapping of DCT coefficient**

Step 5: Modification of the DCT Coefficients

The most effective way of achieving invisibility and low compression ratio, is through embedding each of the watermarked pixel by modifying the polarity between the corresponding pixels on the neighboring blocks. The drawback will be the lack of robustness towards the higher compression ratio attacks. [24]



Step 6: Embedding Into the Relationship between Neighboring Blocks

The next step will be using a 2D residual mask to calculate the polarity of those chosen middle-frequency coefficients between the neighboring blocks.

$P = \text{Polarity}(Y_r)$

$$P = \{p(k \times (M_1 \times \frac{8}{N_1}) + i, l \times (M_2 \times \frac{8}{N_2}) + j),$$

$$0 \leq k < \frac{N_1}{8}, 0 \leq l < \frac{N_2}{8}, 0 \leq i < (M_1 \times \frac{8}{N_1}),$$

$$0 \leq j < (M_2 \times \frac{8}{N_2})\} \quad [30]$$

Where

$$p(k \times (M_1 \times \frac{8}{N_1}) + i, l \times (M_2 \times \frac{8}{N_2}) + j)$$

$$= \begin{cases} 1, & \text{if } y_r(k \times (M_1 \times \frac{8}{N_1}) + i, l \times (M_2 \times \frac{8}{N_2}) + j) \\ & > y_r((k-1) \times (M_1 \times \frac{8}{N_1}) + i, (l-1) \times (M_2 \times \frac{8}{N_2}) + j) \\ 0, & \text{otherwise} \end{cases} \quad [31]$$

Step 7: Reverse the Corresponding Polarity

DCT coefficients of each marked pixels is modified according to the residual mask to reverse the corresponding polarity.

$$\acute{P} = \text{XOR}(P, W_b)$$

$$\acute{P} = \{\acute{p}(i, j), 0 \leq i < M_1 \text{ and } 0 \leq j < M_2\} \quad [32]$$

Where

$$\acute{p}(i, j) = \begin{cases} 1 - p(i, j), & \text{if } w_b(i, j) = 1 \\ p(i, j), & \text{if } w_b(i, j) = 0 \end{cases} = p(i, j) \oplus w_b(i, j) \quad [33]$$

And construct  $\hat{Y}_r$  from  $\acute{P}$

$$\hat{Y}_r = \text{Expand}(\acute{P})$$

$$\text{Such that } \sum_{i,j} (y_r(i, j) - \hat{y}_r(i, j))^2 < \text{threshold} \quad [34]$$

Step 8: Embedding into relationship within each block

The more reliable DC coefficient is used as reference value for each block to solve the propagation of modifications into neighboring blocks.

$$\begin{aligned}
 P &= \text{Polarity } (Y_r) \\
 P &= \{p(k \times (M_1 \times \frac{8}{N_1}) + i, l \times (M_2 \times \frac{8}{N_2}) + j), \\
 &\quad 0 \leq k < \frac{N_1}{8}, 0 \leq l < \frac{N_2}{8}, 0 \leq i < (M_1 \times \frac{8}{N_1}), \\
 &\quad 0 \leq j < (M_2 \times \frac{8}{N_2})\} \quad [30]
 \end{aligned}$$

Where

$$\begin{aligned}
 &p(k \times (M_1 \times \frac{8}{N_1}) + i, l \times (M_2 \times \frac{8}{N_2}) + j) \\
 &= \begin{cases} 1, \text{ if } [ \frac{y_r(k \times (M_1 \times \frac{8}{N_1}) + i, l \times (M_2 \times \frac{8}{N_2}) + j)}{Q(i, j)} ] Q(i, j) \\ > [ \frac{|y(k \times 8, l \times 8)|}{\text{scale factor} \times Q(0,0)} ] Q(0, 0) \\ 0, \text{ otherwise} \end{cases}
 \end{aligned}$$

Step 9: Inverse Block Transform

Lastly, the modified middle-frequency coefficient  $\hat{Y}_r$ , is being mapped into  $Y$  to obtain  $\hat{Y}$ , follows by inverting the associated result (IDCT) in order to retrieve the embedded image.

$$\hat{X} = \text{IDCT}(\hat{Y})$$

### 3.1.2 Watermark Extraction Using DCT Method [24]

For DCT method, extraction of watermark needs its original image, the watermarked image, and either the watermark or the permutation mapping used.

#### Step 1: Block Transform

The original image  $X$  and the suspected image  $\hat{X}$  are both DCT transformed.

$$Y = \text{FDCT}(X)$$

$$\hat{Y} = \text{FDCT}(\hat{X})$$

#### Step 2: Generation of Polarity Patterns

The reduced image is then generated and the middle-frequency DCT coefficients are used to retrieve its polarity patterns.

$$Y_r = \text{Reduce}(Y)$$

$$\hat{Y}_r = \text{Reduce}(\hat{Y})$$

Hence

$$P = \text{Polarity}(Y_r)$$

$$\hat{P} = \text{Polarity}(\hat{Y}_r)$$

#### Step 3: Extraction of The Permuted Data

XOR is performed on the two polarity patterns in order to obtain a permuted binary data.

$$\hat{W}_b = \text{XOR}(P, \hat{P})$$

Where

$$\hat{w}_b(i, j) = p(i, j) \oplus \hat{p}(i, j)$$

#### Step 4: Reverse Block-Based Image-Dependent Permutation

$\hat{W}_b$  is reverse permuted to obtain  $\hat{W}_p$ .

#### Step 5: Reverse Pseudorandom Permutation

The reverse permutation process is repeated on  $\hat{W}_p$  to retrieve back the watermark  $\hat{W}$ .

$$\hat{w}(i, j) = \hat{w}_p(i', j')$$

Step 6: Similarity Measurement

The similarity between referenced watermark  $W$  and extracted watermark  $\hat{W}$  can be measured using the cross-relation normalized by the reference watermark energy to give unity as the peak correlation. [24]

$$\text{Normalized Correlation (NC)} = \frac{\sum_i \sum_j W(i,j) \hat{W}(i,j)}{\sum_i \sum_j [W(i,j)]^2}$$

### 3.1.3 DCT Watermark Embedding Flowchart [24]

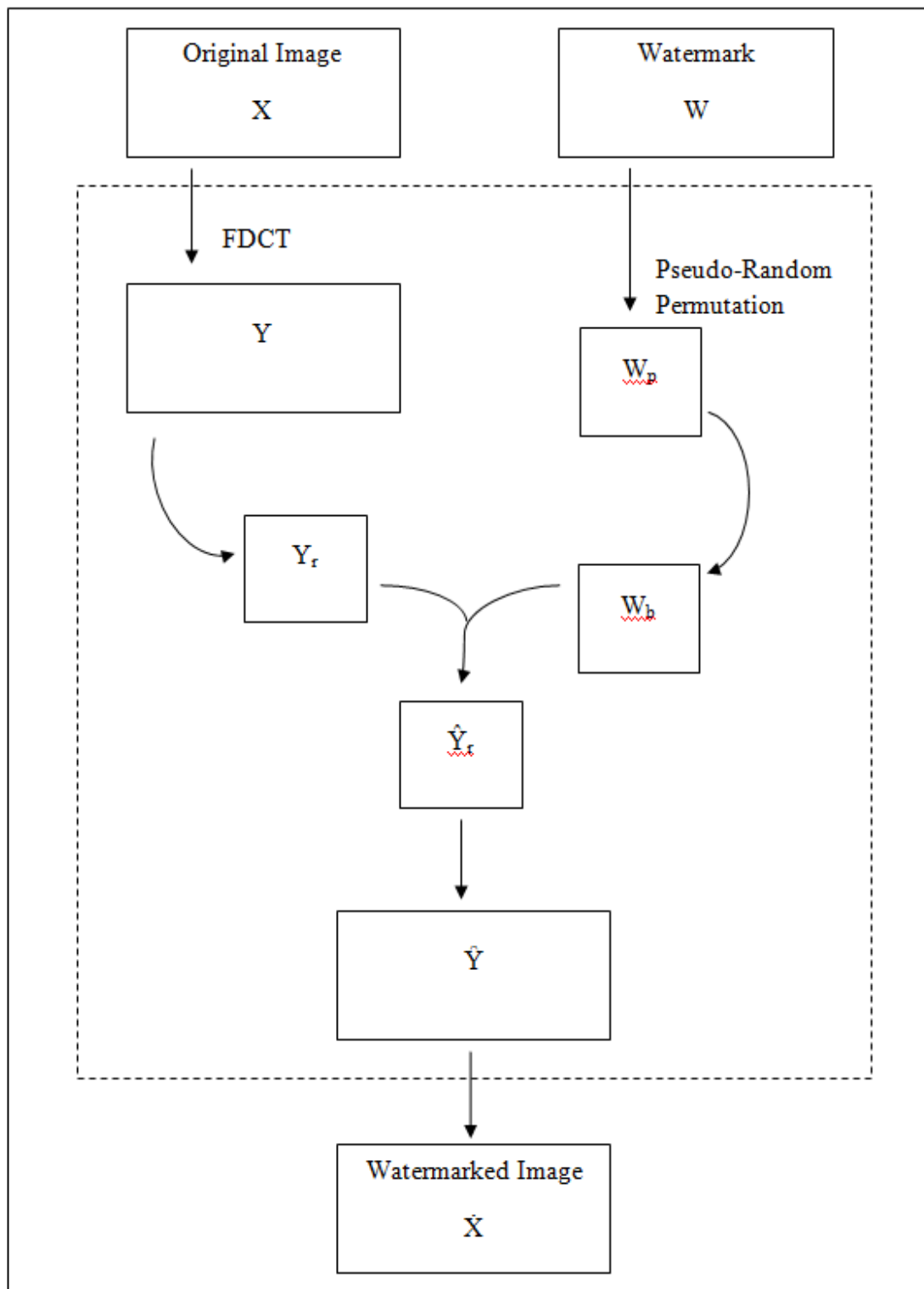


Figure 3.3 DCT Watermark Embed Flowchart

### 3.1.4 DCT Watermark Extraction Flowchart [24]

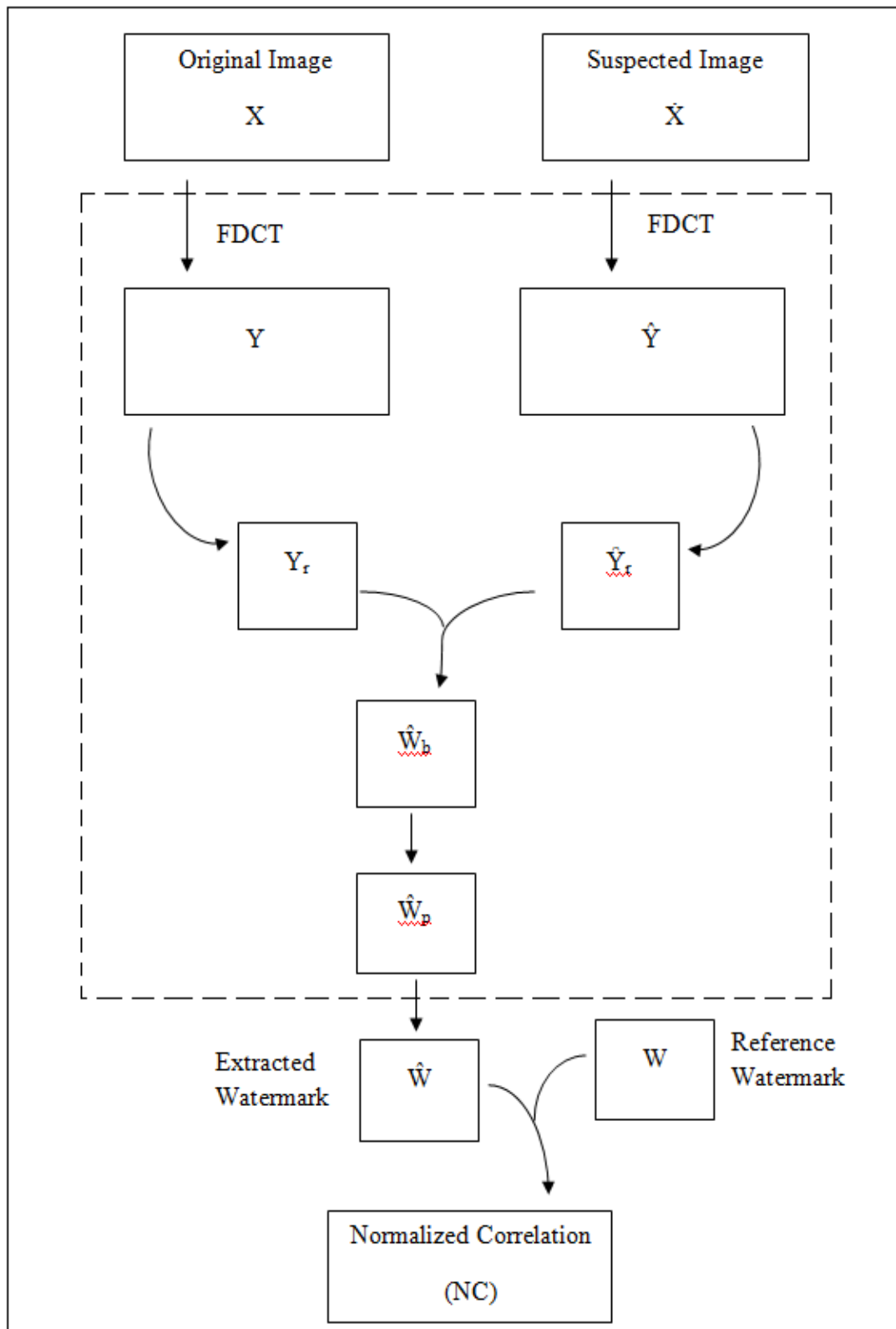


Figure 3.4 DCT Watermark Extract Flowchart

## 3.2 DWT [11]

### 3.2.1 Watermark Embedding Using DWT Method [35]

#### Step 1: Improving Robustness of Watermark Algorithm

The original image is DCT transformed in order to obtain a disordered image.

#### Step 2: DWT Transform

The host image  $X$  is decomposed by  $L$ -levels using two-dimensional DWT. Hence, a approximating sub-image and  $3L$  detail sub-images are obtained. The level of DWT will affect the concealing effect of embedding watermark.

#### Step 3: Choose the Streak Blocks

All the high frequency band information of DWT is being potted into  $2 \times 2$  image sub-blocks  $B_k$ . Then the entropy and square values of each  $B_k$  is calculated. The streak blocks wanted,  $U_k(k = 1, 2, \dots, P \times Q)$  can be obtained by selecting the appropriate threshold of entropy and square.

#### Step 4: Embedding The Watermark

The wavelet coefficient values,  $C_k$ , of the chosen streak blocks,  $B_k$  are altered to complete the watermark embedding process.

$$C_k' = C_k + a \times v_k, k = 1, 2, \dots, P \times Q \quad [36]$$

#### Step 5: Inversing Transform

Lastly, all the information of lowest frequency band and the mended high frequency band are combined, before inversing by  $L$ -level, to obtain the watermarked image.

### **3.2.2 Watermark Extraction Using DWT Method [35]**

#### Step 1: DWT Transform

Both the original image and watermarked image are transformed by L-levels using DWT, in order to gain the information of lowest and highest frequency band.

#### Step 2: Make Sure the Streak Blocks

The high-frequency band information of both original and watermarked image are plotted into 2 x 2 image sub blocks. The streak block,  $U$ , is used as index and  $U'$  of the corresponding sub block of DWT transformed watermarked image is obtained.

#### Step 3: Distilling the Watermarking Signal $V'$

The entropies  $H(U_k)$  and  $H(U_k')$  are calculated and the result of  $H(U_k) - H(U_k')$  is acquired. If the value is larger than a certain threshold value, then it's signed as 1, else, it will be signed as 0.

#### Step 4: Inverse Transformation of Watermark

By inverse DCT of the disordered watermarking image, the watermark image is retrieved.



### 3.2.3 DWT Watermark Embedding Flowchart [35]

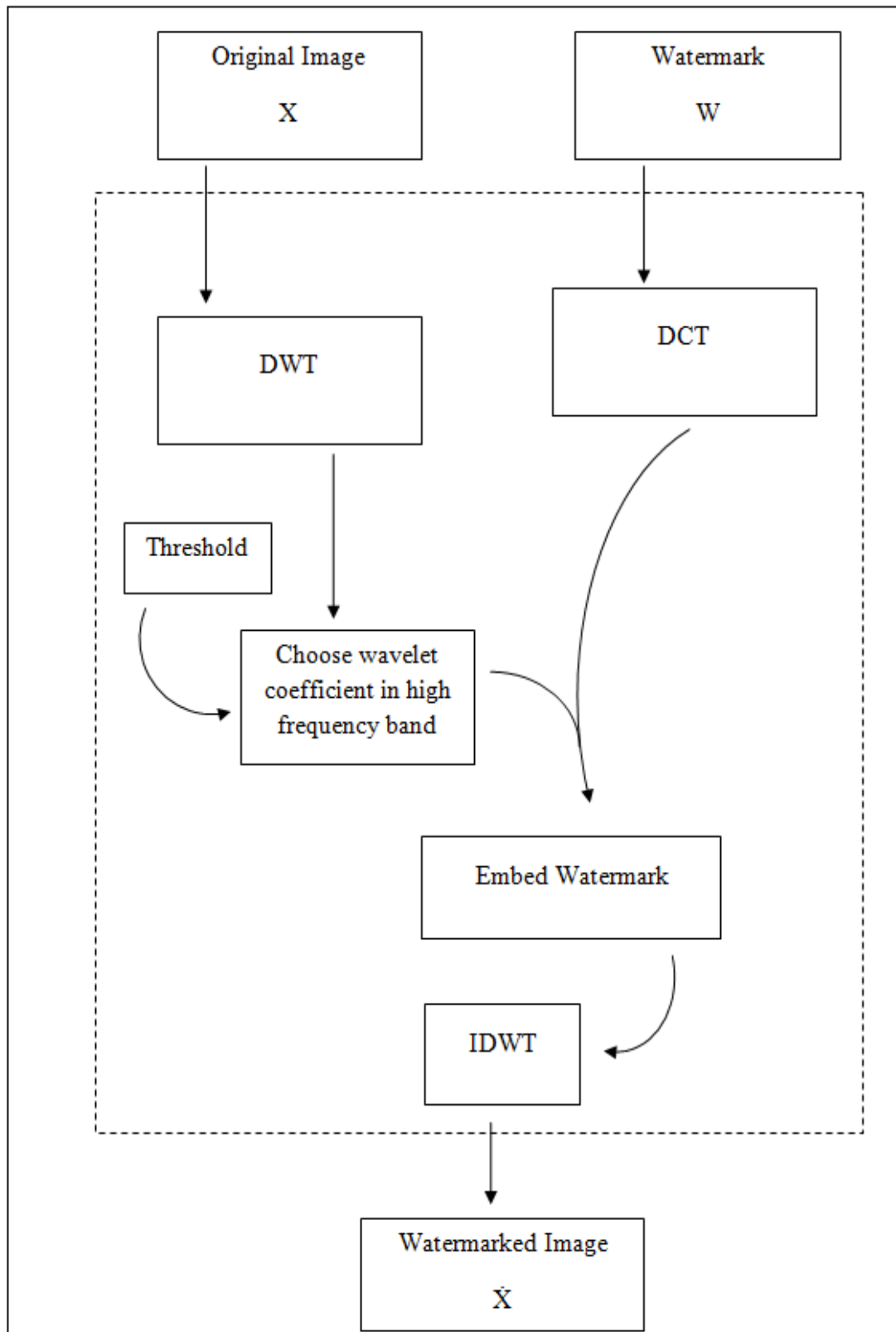


Figure 3.5 DWT Watermark Embed Flowchart

### 3.2.4 DWT Watermark Extraction Flowchart [35]

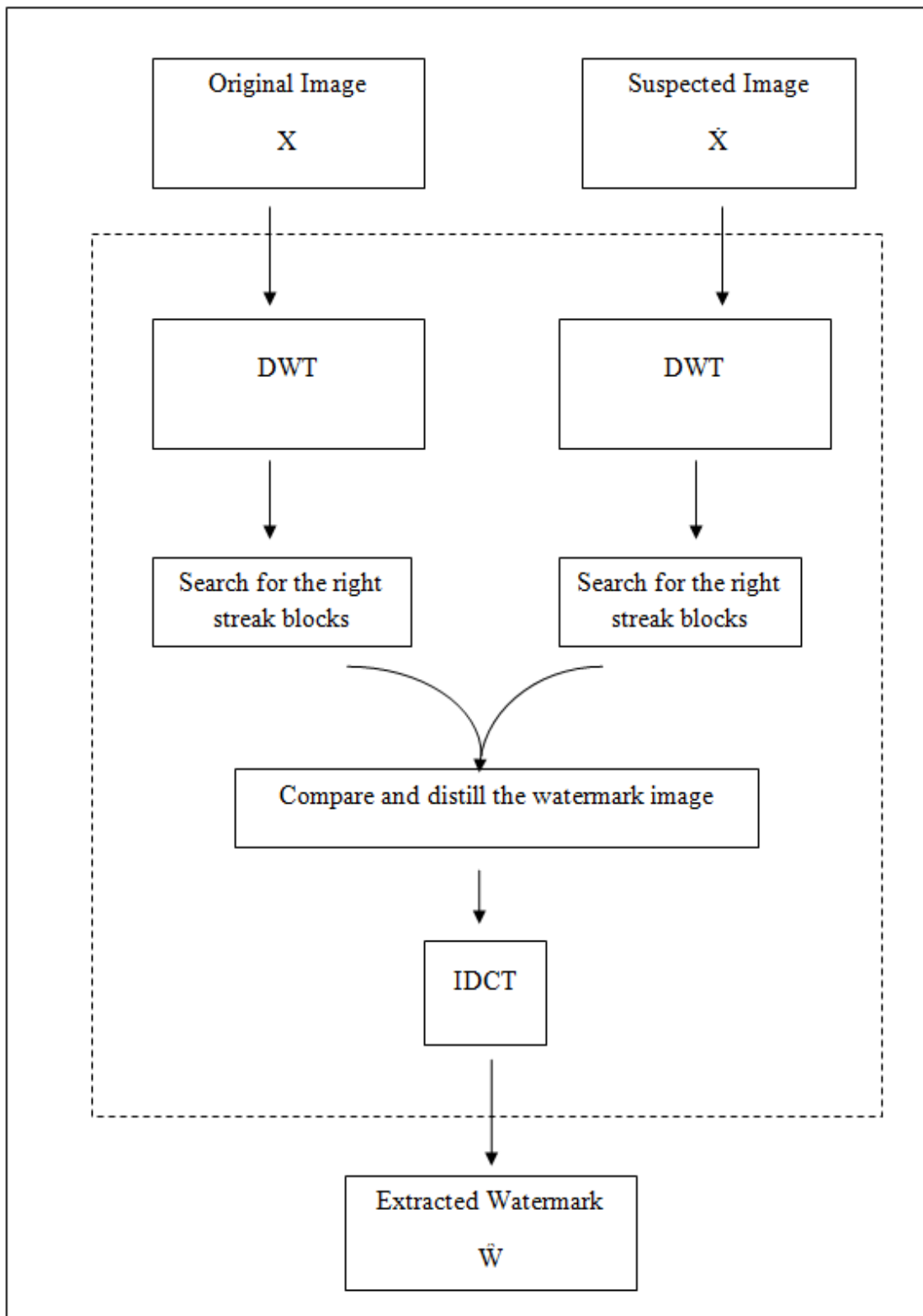


Figure 3.6 DWT Watermark Extract Flowchart

### 3.3 Spatial Domain

#### 3.3.1 Watermark Embedding Using Spatial Domain Method [37]

The original watermark is encrypted with secret key using XOR operation, before going through pseudorandom permutation to obtain a permuted watermark. Meanwhile, for the host image, it's separated into Red, Green and Blue layers. The blue layer is taken out and divided into 16 regions of 128 x 128, hence scrambled and shuffled before the embedding process. After the watermark is embedded, each region of 128 x 128 is inverse scrambled before recombine with the R and G channel to get the watermarked image.

#### 3.3.2 Watermark Extraction Using Spatial Domain Method [37]

For the extraction process using spatial domain method, the original image and original watermark are compulsory. First of all, the Blue (B) channel of both watermarked image and host image are extracted, and then each region of 128 x 128 is scrambled. By comparing the intensity pixel values of each region in original image with the corresponding watermarked image, the permuted watermark is extracted. Hence, inverse pseudorandom permutation will take place before performing XOR operation with the secret key used in embedding process, to acquire the extracted watermark. Next will be the comparison process between the extracted watermark with the original watermark. The normalized cross correlation (NCC) between the original watermark and extracted watermark is calculated as follows:

$$NCC = \frac{\sum_i \sum_j W(i,j) \hat{W}(i,j)}{\sum_i \sum_j [W(i,j)]^2}$$

### 3.3.3 Spatial Domain Watermark Embedding Flowchart [37]

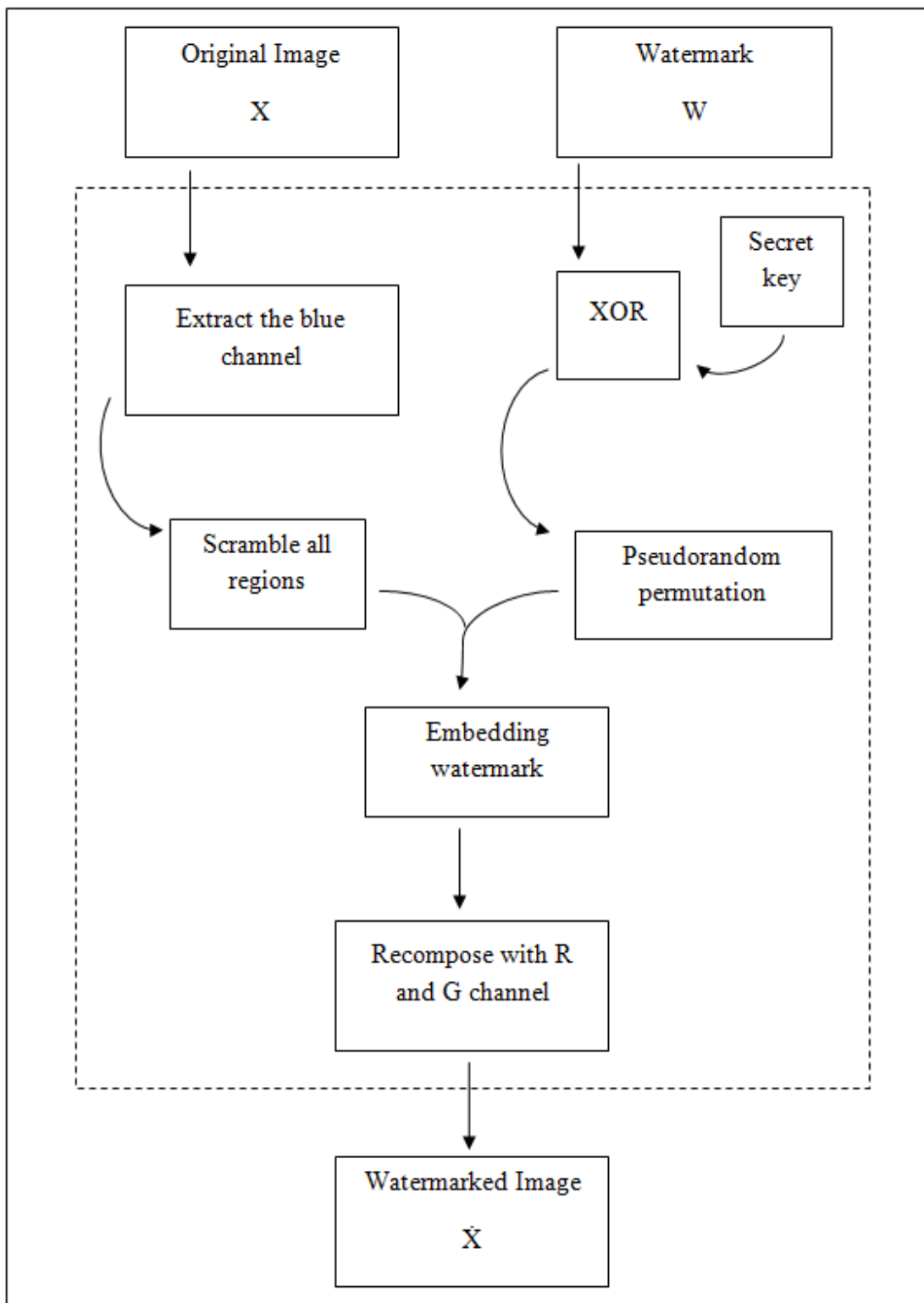


Figure 3.7 Spatial Domain Embed Flowchart

### 3.3.4 Spatial Domain Watermark Extraction Flowchart [37]

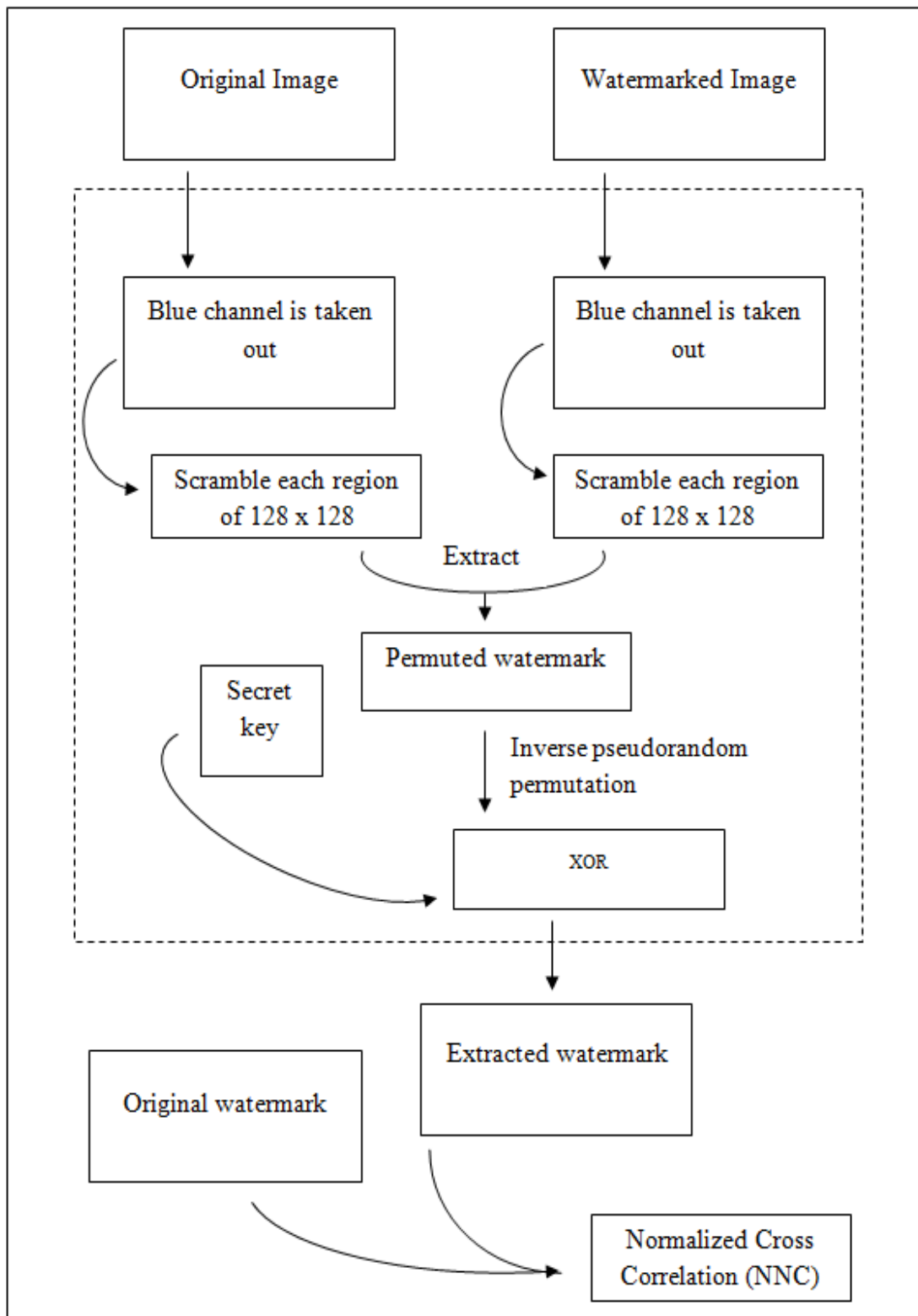


Figure 3.8 Spatial Domain ExtractFlowchart

### **3.4 Timeline**

Time management is the most important factor of completing a project within the given time. As for this Final Year Project 1, the timeline can be categorized into 4 main phases as shown below:

#### **3.4.1 Phase 1: Literature Review Phase**

Throughout this phase, research papers, journals and survey papers which are related to digital watermarking are deliberated and analyzed in order to gain more knowledge and information about this field. Different algorithms of watermarking are studied and compared.

Time frame: 6 June 2010 to 4 July 2010

#### **3.4.2 Phase 2: Research Methodology Phase**

For this phase, DCT, DWT and spatial domain methods are studied in detail and the flow of each algorithm is explained and sketched. The advantages and disadvantages of spatial domain and transform domain are determined.

Time duration: 4 July 2010 to 7 August 2010

#### **3.4.3 Phase 3: Development Phase**

After completing the methodology phase, all the algorithms chosen are developed using MATLAB. Hence, Sitrmark will be used to perform attacks on the watermarks.

Time duration: 1 October 2010 to 1 March 2011

### 3.4.4 Phase 4: Adjustment and Documentation Phase


Next on the list will be adjustment and documentation. Minor flaws and disfigurement of the application are adjusted and fixed before being documented into the final report.

Time duration: 1 March 2011 to 1 April 2011

### 3.4.5 Milestone

Year / Month	2010					2011			
Activities	June	July	Aug	Sept	Oct	Jan	Feb	Mac	Apr
Search for related papers									
Research about watermarking									
Study and evaluate algorithms									
Collect and formulate idea for selected algorithm									
Develop algorithms using MATLAB									
Benchmark of every algorithms									
Documentation and final report compilation									
Final adjustment									

 Completed

 To be complete

**Table 3.1 Milestone**

### **3.5 Development Tools**

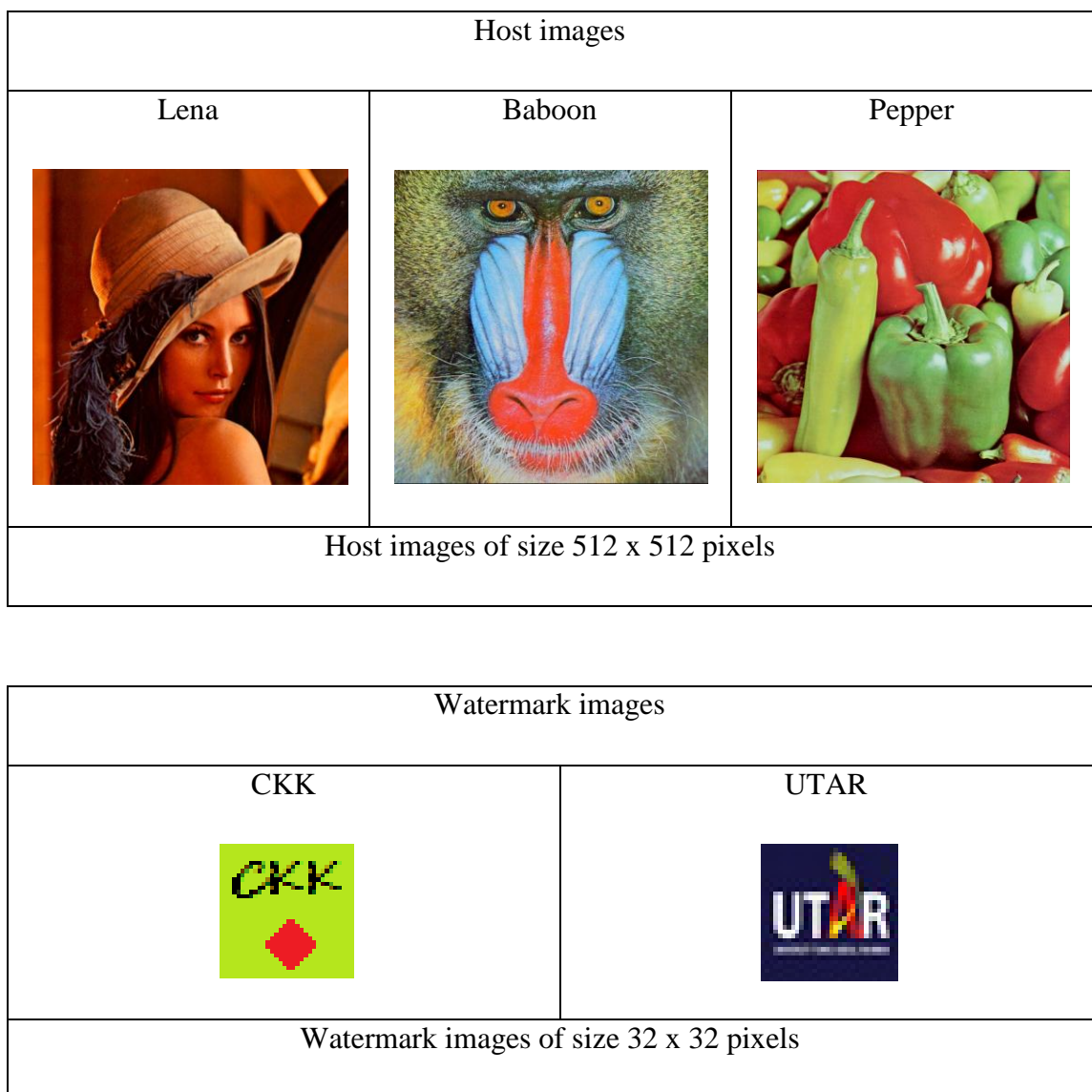
As for this final year project, MATLAB has been chosen as the primary tool for development and implementation process. MATLAB can be easily accessed in Universiti Tunku Abdul Rahman (UTAR) as UTAR purchased 10 MATLAB licenses for students' study and research purpose. Hence, software availability and piracy problems are solved.

MATLAB is a powerful yet effective and interactive tool which provides users with different functions to solve their problems, such as algorithm development, data analysis and visualization, numeric computation, graphical user interface and other useful functions. MATLAB will be used for development and implementation of the algorithms chosen for this project, which are spatial-based and DCT-based.



## Chapter 4 SIMULATIONS AND RESULTS

For the simulation process, both the DCT and Spatial domain will undergo series of tests in order to compare their robustness and quality of watermarking process. In addition, 60 different attacks are applied to the watermarked images, which include noise addition, filtering, compression, rotation, image cropping, self similarities tests and print screen attack. The opted host and watermark images are as follows:



**Figure 4.1 Host and watermark images**

## 4.1 Limitations

Both the DCT-based and spatial-based watermarking are being implemented effectively and able to perform wisely. However, there are several drawbacks in them as such limitations exist during the coding implementation stage. Below are some of the limitations of the algorithms:

### ❖ Non-blind watermarking algorithm

Both the host image and watermark are compulsory for the extraction process. If one of the images happens to be missing or corrupted, the detection process can't take place as these images are required for reference.

### ❖ Size

Size of the host images and watermark images are one of the limitations as well. Both the algorithms will only accept host image of size 512 x 512 pixels, and watermark image of size 32 x 32 pixels. Any picture larger or smaller will cause the embed and extract process fail to perform flawlessly.

### ❖ Binary image

For spatial-based watermarking, only binary watermark are allowed to be embedded into host images. The algorithm will convert the watermark to binary if the selected watermark happens to be a coloured watermark, where the converted watermark is in black and white ('0' and '1').

## 4.2 Graphical User Interface (GUI)

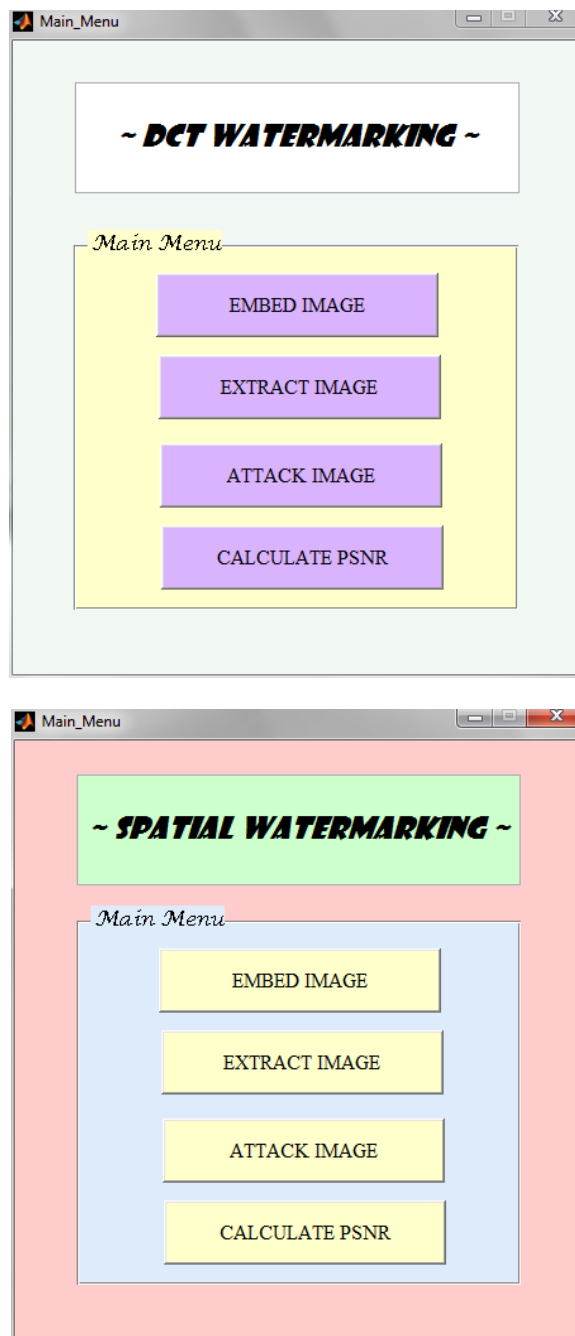


Figure 4.2 Main menu for DCT and spatial domain watermarking

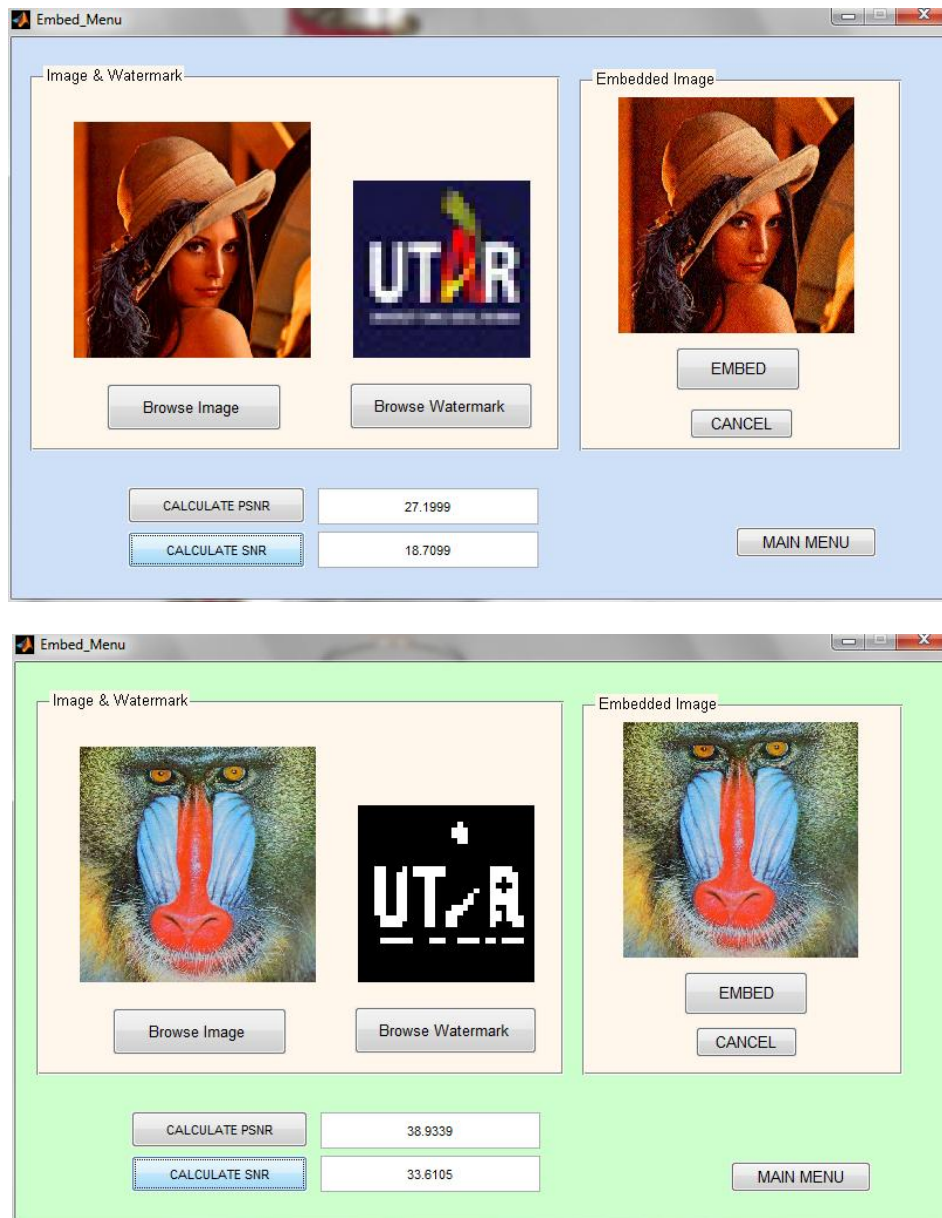


Figure 4.3 Embed menu for DCT and spatial domain watermarking

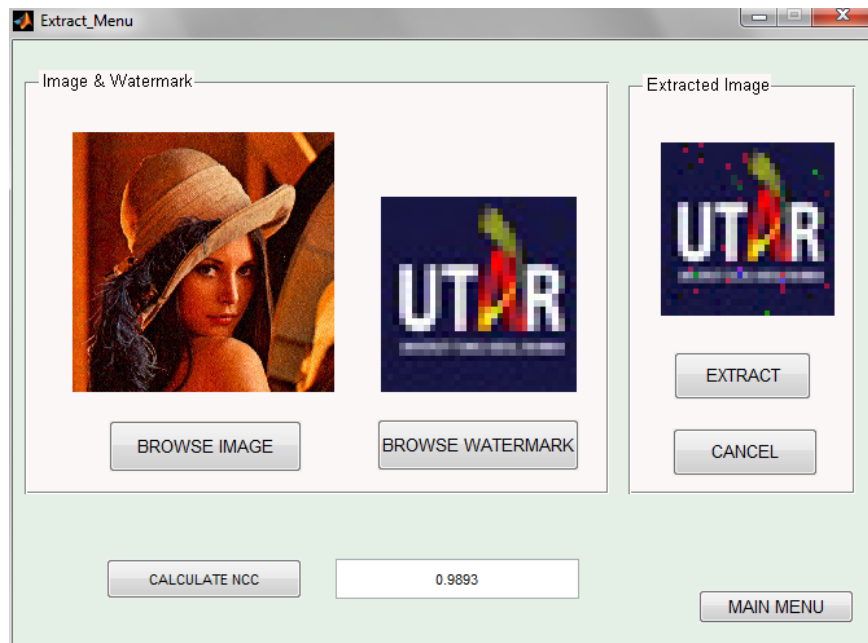
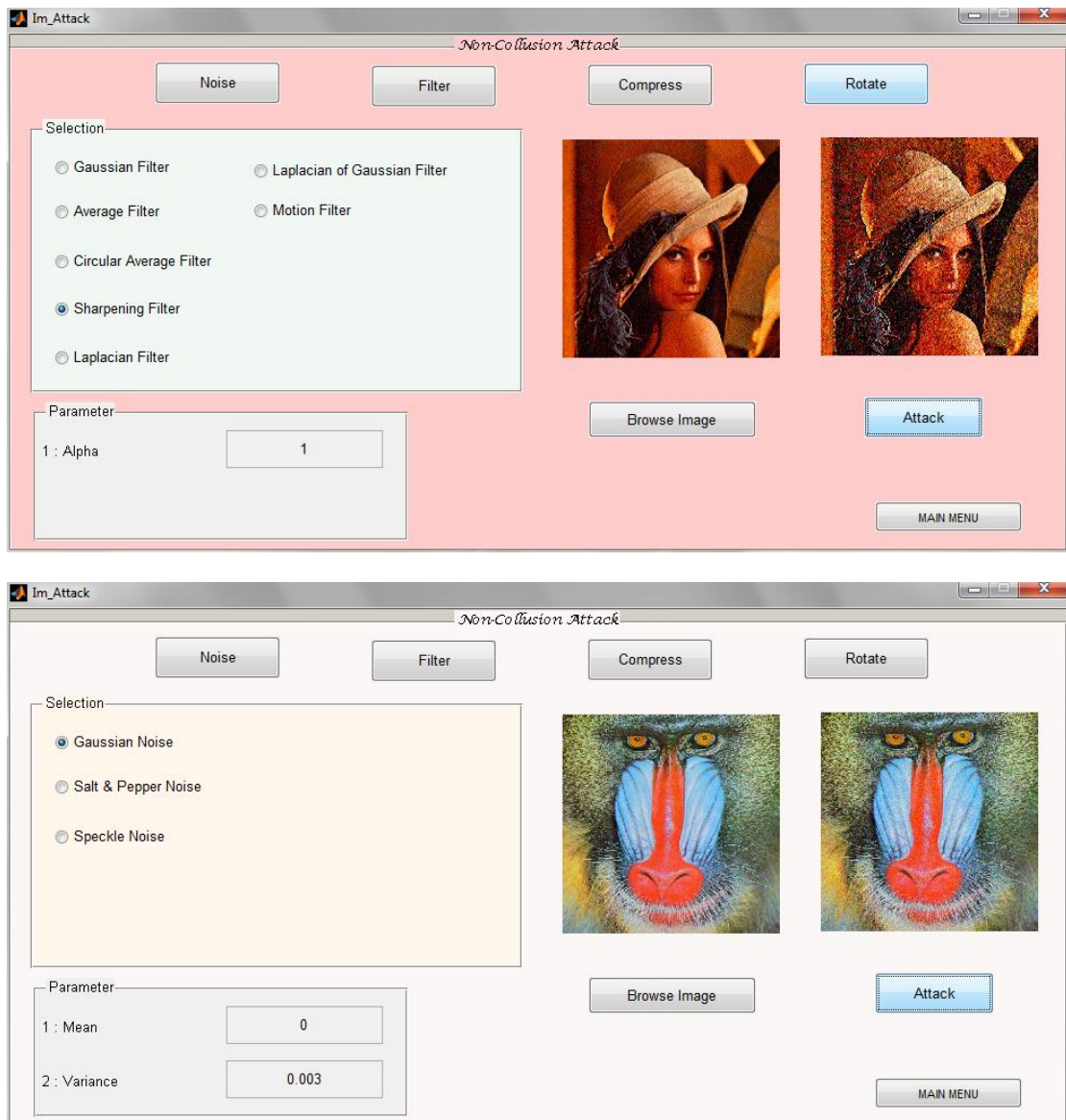
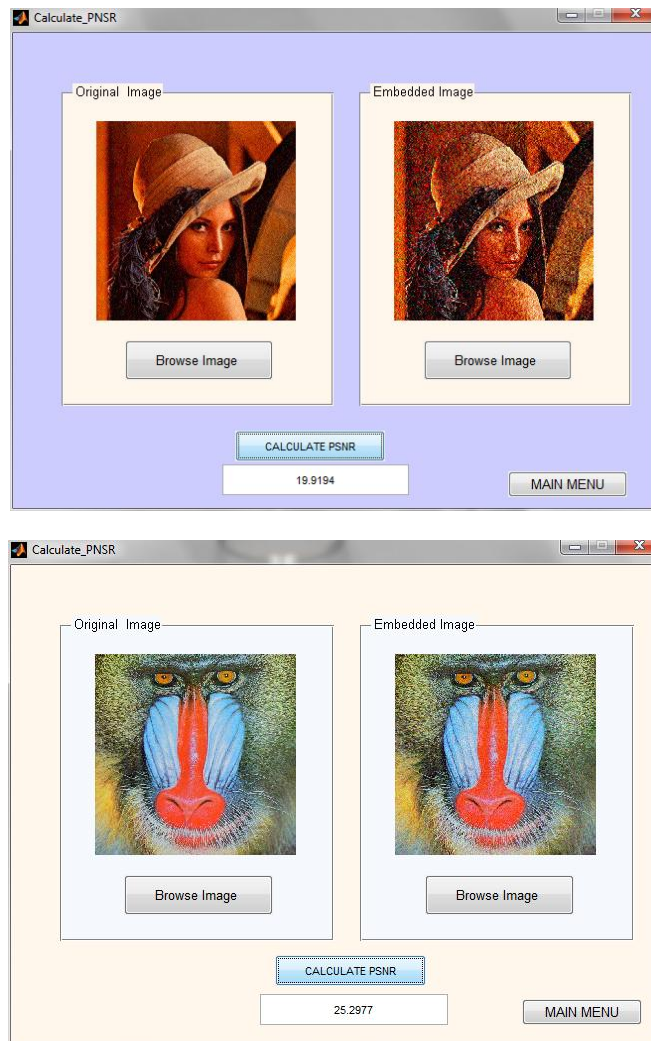


Figure 4.4 Extract menu for DCT and spatial domain watermarking



**Figure 4.5** Attack menu for DCT and spatial domain watermarking



**Figure 4.6 PSNR calculation menu for DCT and spatial domain watermarking**

## 4.3 Simulations

### 4.3.1 Simulation Set 1: Attack Free

As for this set of simulation, Lena, Baboon and Pepper images are used as host images for CKK and UTAR watermark. Both the CKK and UTAR watermark are embedded into the selected host images using DCT algorithm and spatial algorithm. This simulation set is carried out to obtain the Peak Signal to Noise Ratio (PSNR) between host images and watermarked images. Moreover, the Correlation Coefficient Value (NCC) of the extracted watermark will be calculated as well.



### 4.3.2 Simulation Set 2: CKK Watermark

Meanwhile, series of non-collusion attacks are applied towards the CKK watermarked images, alongside with image cropping operation, print screen attack and self similarities tests. All the attacks can be summarized as follows:

Type of attacks	Attack name ( <i>parameter</i> )
Noise addition	Gaussian noise ( <i>mean, variance</i> ) Salt & pepper ( <i>noise intensity</i> ) Speckle noise ( <i>variance</i> )
Image filtering	Gaussian filter ( <i>size, standard deviation</i> ) Average filter ( <i>size</i> ) Circular average filter ( <i>radius</i> ) Sharpening filter ( <i>alpha</i> ) Laplacian filter ( <i>alpha</i> ) Laplacian of Gaussian filter ( <i>hsize, sigma</i> ) Motion filter ( <i>len, theta</i> )
Compression	JPEG compression ( <i>percentage</i> )
Rotation	Bilinear rotation ( <i>degree</i> ) Bicubic rotation ( <i>degree</i> ) Nearest rotation ( <i>degree</i> )
Crop	Image crop ( <i>percentage</i> )
Print screen	Duplication of image using print screen function
Self similarities	Self similarities ( <i>colour space, channel, type, percentage</i> )

**Table 4.1 Attacks for CKK watermark**

### 4.3.3 Simulation Set 3: UTAR Watermark

On the other hand, simulation set 3 will go through the same attacks as simulation set 2, but the watermark will be UTAR logo instead of CKK logo. The attacks applied are listed in the table below:

Type of attacks	Attack name ( <i>parameter</i> )
Noise addition	Gaussian noise ( <i>mean, variance</i> ) Salt & pepper ( <i>noise intensity</i> ) Speckle noise ( <i>variance</i> )
Image filtering	Gaussian filter ( <i>size, standard deviation</i> ) Average filter ( <i>size</i> ) Circular average filter ( <i>radius</i> ) Sharpening filter ( <i>alpha</i> ) Laplacian filter ( <i>alpha</i> ) Laplacian of Gaussian filter ( <i>hsize, sigma</i> ) Motion filter ( <i>len, theta</i> )
Compression	JPEG compression ( <i>percentage</i> )
Rotation	Bilinear rotation ( <i>degree</i> ) Bicubic rotation ( <i>degree</i> ) Nearest rotation ( <i>degree</i> )
Crop	Image crop ( <i>percentage</i> )
Print screen	Duplication of image using print screen function
Self similarities	Self similarities ( <i>colour space, channel, type, percentage</i> )

**Table 4.2 Attacks for UTAR watermark**

## 4.4 Simulation Result

### 4.4.1 Simulation Set 1 Result

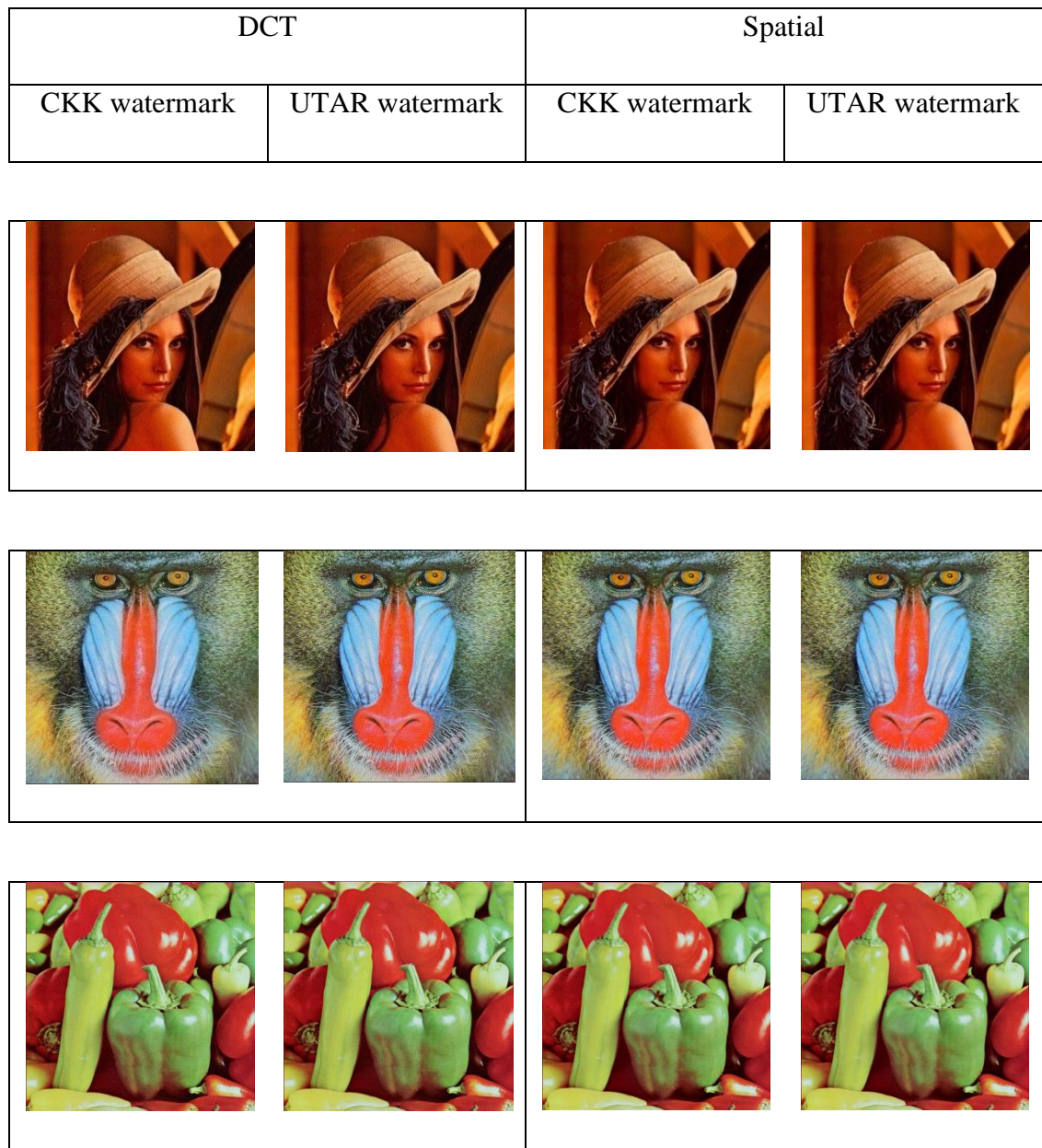


Figure 4.7 Watermarked images of size 512 x 512 pixels

Watermark \ Image		SNR		
		Lena	Baboon	Pepper
CKK logo	DCT	18.2843	19.932	20.3727
	Spatial	31.7423	33.6104	33.1451







Watermark \ Image		SNR		
		Lena	Baboon	Pepper
UTAR logo	DCT	18.7099	20.5452	18.7099
	Spatial	31.6605	33.6105	33.1358







**Table 4.3 SNR comparisons between DCT and spatial algorithm watermarked images**

Watermark \ Image		PSNR		
		Lena	Baboon	Pepper
CKK logo	DCT	26.7742	25.2554	26.297
	Spatial	40.2323	38.9338	39.0693

Watermark \ Image		PSNR		
		Lena	Baboon	Pepper
UTAR logo	DCT	27.1999	25.8686	27.1999
	Spatial	40.1505	38.9339	39.06

**Table 4.4 PSNR comparisons between DCT and spatial algorithm watermarked images**

Watermark \ Image		NCC		
		Lena	Baboon	Pepper
CKK logo	DCT	 0.97436	 0.99009	 1
	Spatial	 1	 1	 1

Watermark \ Image		NCC		
		Lena	Baboon	Pepper
UTAR logo	DCT	 0.9893	 0.99684	 1
	Spatial	 1	 1	 1

**Table 4.5 NCC comparisons between DCT and spatial algorithm watermarked images**



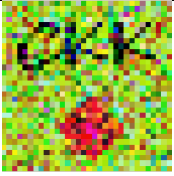



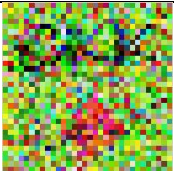

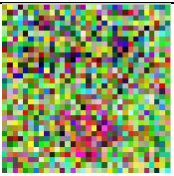

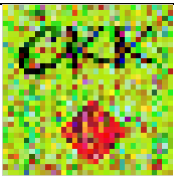

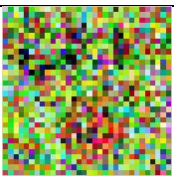

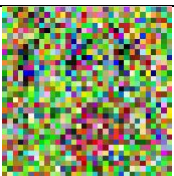



### 4.4.2 Simulation Set 2 Results

**Table 4.6 PSNR comparison for Lena and CKK watermarked images**



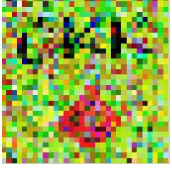

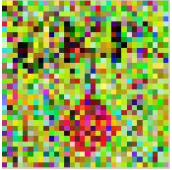

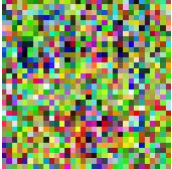

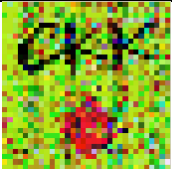

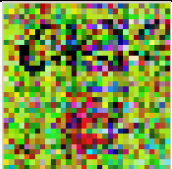

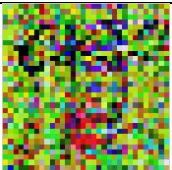

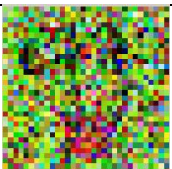

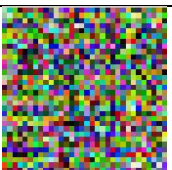

No.	Attack	PSNR	
		DCT	Spatial
1	Gaussian noise (0, 0.001)	30.6043	30.6462
2	Gaussian noise (0, 0.003)	26.0135	26.0396
3	Gaussian noise (0, 0.005)	23.8871	23.9141
4	Gaussian noise (0, 0.01)	21.0651	21.0957
5	Gaussian noise (0, 0.03)	16.6608	16.6826
6	Salt & pepper noise (0.01)	24.1421	24.1722
7	Salt & pepper noise (0.05)	17.2273	17.1613
8	Salt & pepper noise (0.1)	14.1435	14.1486
9	Speckle noise (0.01)	28.7747	28.7963
10	Speckle noise (0.05)	22.0724	22.0936
11	Speckle noise (0.1)	19.2593	19.3018
12	Speckle noise (0.5)	13.2121	13.2789
13	Gaussian filter (3, 1)	27.8368	34.2019
14	Gaussian filter (3, 2)	26.7874	33.1081
15	Gaussian filter (3, 3)	26.6226	32.932
16	Gaussian filter (4, 1)	25.6804	30.3214
17	Gaussian filter (4, 2)	24.9409	29.6256
18	Average filter (3)	26.4955	32.8005
19	Average filter (4)	24.6689	29.3473
20	Circular average filter (1)	30.1117	37.4025
21	Circular average filter (1.5)	27.509	34.8868
22	Circular average filter (1.7)	26.6972	33.9021
23	Circular average filter (2)	25.7995	32.5614
24	Sharpening filter (0.1)	17.4878	24.2838
25	Sharpening filter (0.3)	18.2345	25.1802
26	Sharpening filter (0.5)	18.8016	25.8324
27	Sharpening filter (1)	19.7145	26.7938
28	Laplacian filter (0.2)	8.9137	8.712
29	Laplacian filter (0.5)	8.8934	8.6895
30	Laplacian filter (0.7)	8.8817	8.6802
31	Laplacian filter (1)	8.8676	8.67
32	Laplacian of Gaussian filter (5, 0.5)	8.8546	8.7755

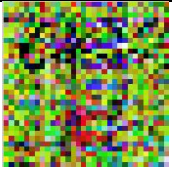

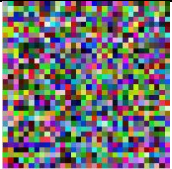



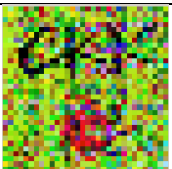

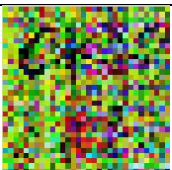

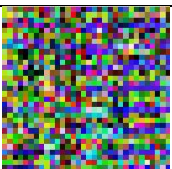







33	Laplacian of Gaussian filter (7, 0.5)	8.8331	8.7665
34	Laplacian of Gaussian filter (10, 0.5)	8.7207	8.7521
35	Laplacian of Gaussian filter (5, 0.7)	8.7584	8.6204
36	Laplacian of Gaussian filter (5, 1.0)	8.5699	8.552
37	Motion filter (9, 45)	25.1338	29.2436
38	Motion filter (9, 90)	25.2403	29.4362
39	JPEG compression (45)	27.0052	32.3415
40	JPEG compression (90)	29.5804	36.2358
41	JPEG compression (95)	30.153	37.7539
42	Bilinear rotation (0.1)	31.88	37.5187
43	Bilinear rotation (0.2)	26.4666	31.8527
44	Bilinear rotation (0.3)	23.999	28.8623
45	Bilinear rotation (0.4)	23.9997	27.0746
46	Bicubic rotation (0.1)	23.9997	38.195
47	Bicubic rotation (0.2)	26.0217	31.6283
48	Bicubic rotation (0.3)	23.504	28.6021
49	Nearest rotation (0.1)	Inf	Inf
50	Nearest rotation (0.2)	24.2497	29.8682
51	Nearest rotation (0.3)	22.7113	27.5616
52	Image crop (25)	10.0419	10.1147
53	Image crop (50)	10.4403	10.5234
54	Image crop (75)	11.8891	12.0155
55	Print screen	Inf	Inf
56	Rotation (45)	9.4242	9.4673
57	Rotation (90)	9.9815	10.0631
58	Self similarities (hsv, 001, s, 60)	23.9564	25.6733
59	Self similarities (rgb, 001, s, 60)	25.2477	25.6915
60	Self similarities (yuv, 100, s, 60)	23.9264	25.3926



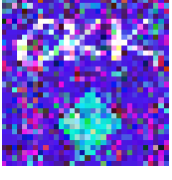

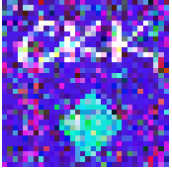

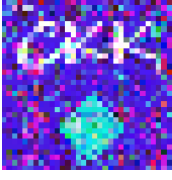

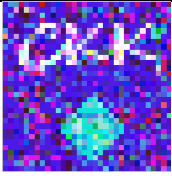

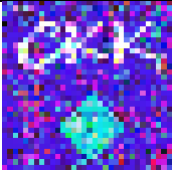

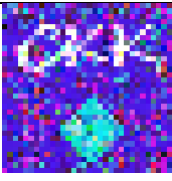



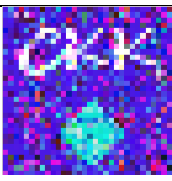

**Table 4.7 NCC comparison for Lena and CKK watermarked images**

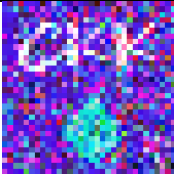

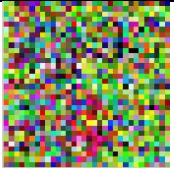

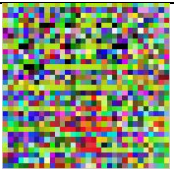

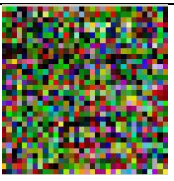

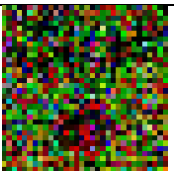

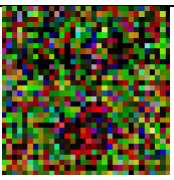



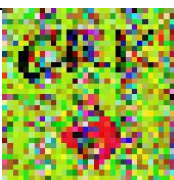

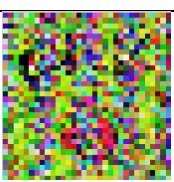

No.	Attack	DCT		Spatial	
			NCC		NCC
1	Gaussian noise (0, 0.001)		0.91798		0.989437
2	Gaussian noise (0, 0.003)		0.81828		0.92723
3	Gaussian noise (0, 0.005)		0.74967		0.893192
4	Gaussian noise (0, 0.01)		0.64466		0.847418
5	Gaussian noise (0, 0.03)		0.44058		0.715962
6	Salt & pepper noise (0.01)		0.85694		1
7	Salt & pepper noise (0.05)		0.53609		1
8	Salt & pepper noise (0.1)		0.42996		1





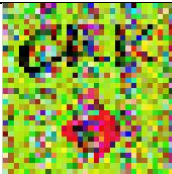

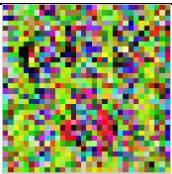





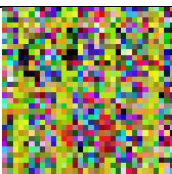







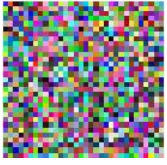













9	Speckle noise (0.01)		0.90058		1
10	Speckle noise (0.05)		0.76022		1
11	Speckle noise (0.1)		0.66473		1
12	Speckle noise (0.5)		0.40732		1
13	Gaussian filter (3, 1)		0.83247		0.588028
14	Gaussian filter (3, 2)		0.62196		0.562207
15	Gaussian filter (3, 3)		0.57526		0.557512
16	Gaussian filter (4, 1)		0.56117		0.551643
17	Gaussian filter (4, 2)		0.16897		0.529343

18	Average filter (3)		0.51843		0.548122
19	Average filter (4)		-0.012761		0.515258
20	Circular average filter (1)		0.9172		0.714789
21	Circular average filter (1.5)		0.72247		0.568075
22	Circular average filter (1.7)		0.52791		0.546948
23	Circular average filter (2)		0.026186		0.53169
24	Sharpening filter (0.1)		0.9398		1
25	Sharpening filter (0.3)		0.94452		1
26	Sharpening filter (0.5)		0.94866		1

27	Sharpening filter (1)		0.94745		1
28	Laplacian filter (0.2)		-0.86082		0
29	Laplacian filter (0.5)		-0.8746		0
30	Laplacian filter (0.7)		-0.87485		0
31	Laplacian filter (1)		-0.87632		0
32	Laplacian of Gaussian filter (5, 0.5)		-0.8825		0
33	Laplacian of Gaussian filter (7, 0.5)		-0.88155		0
34	Laplacian of Gaussian filter (10, 0.5)		-0.32249		0.517606
35	Laplacian of Gaussian filter (5, 0.7)		-0.89149		0.00117371

36	Laplacian of Gaussian filter (5, 1.0)		-0.81408		0.480047
37	Motion filter (9, 45)		0.37139		0.5223
38	Motion filter (9, 90)		0.25055		0.529343
39	JPEG compression (45)		0.27351		0.534038
40	JPEG compression (90)		0.46336		0.577465
41	JPEG compression (95)		0.46479		0.615023
42	Bilinear rotation (0.1)		0.93065		1
43	Bilinear rotation (0.2)		0.7206		0.954225
44	Bilinear rotation (0.3)		0.44967		0.812207

45	Bilinear rotation (0.4)		0.18536		0.697183
46	Bicubic rotation (0.1)		0.93708		1
47	Bicubic rotation (0.2)		0.73039		1
48	Bicubic rotation (0.3)		0.4511		0.956573
49	Nearest rotation (0.1)		0.97436		1
50	Nearest rotation (0.2)		0.6735		1
51	Nearest rotation (0.3)		0.41031		1
52	Image crop (25)		-0.0099288		0.510563
53	Image crop (50)		-0.0077223		0.494131

54	Image crop (75)		-0.0001074		0.484742
55	Print Screen		0.97436		1
56	Rotation (45)		-0.037128		0.517606
57	Rotation (90)		-0.0099065		0.503521
58	Self similarities (hsv, 001, s, 60)		0.62137		0.482394
59	Self similarities (rgb, 001, s, 60)		0.97373		0.482394
60	Self similarities (yuv, 100, s, 60)		0.74451		0.482394







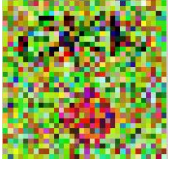
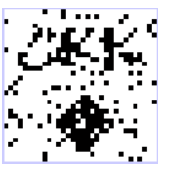




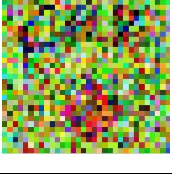

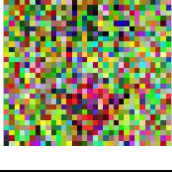



**Table 4.8 PSNR comparison for Baboon and CKK watermarked image**

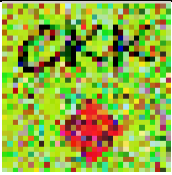

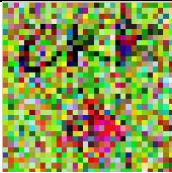

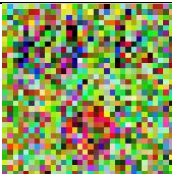

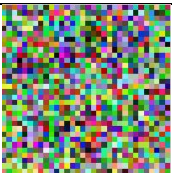

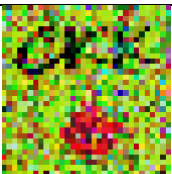

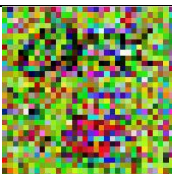

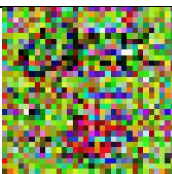

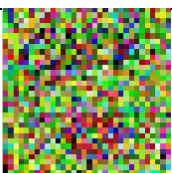

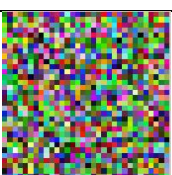

No.	Attack	PSNR	
		DCT	Spatial
1	Gaussian noise (0, 0.001)	30.0543	30.0252
2	Gaussian noise (0, 0.003)	25.3117	25.2885
3	Gaussian noise (0, 0.005)	23.1316	23.0987
4	Gaussian noise (0, 0.01)	20.1953	20.1647
5	Gaussian noise (0, 0.03)	15.7304	15.6956
6	Salt & pepper noise (0.01)	25.1249	25.3202
7	Salt & pepper noise (0.05)	18.2463	18.26
8	Salt & pepper noise (0.1)	15.243	15.2516
9	Speckle noise (0.01)	25.5286	25.5308
10	Speckle noise (0.05)	18.8205	18.8288
11	Speckle noise (0.1)	16.0207	16.0191
12	Speckle noise (0.5)	10.0402	10.0412
13	Gaussian filter (3, 1)	22.5027	24.3358
14	Gaussian filter (3, 2)	21.4727	23.2996
15	Gaussian filter (3, 3)	21.3108	23.1365
16	Gaussian filter (4, 1)	20.7935	22.5123
17	Gaussian filter (4, 2)	20.0317	21.7493
18	Average filter (3)	21.1869	23.0117
19	Average filter (4)	19.7508	21.4604
20	Circular average filter (1)	24.5574	26.4329
21	Circular average filter (1.5)	21.9804	23.8558
22	Circular average filter (1.7)	21.2647	23.1741
23	Circular average filter (2)	20.482	22.396
24	Sharpening filter (0.1)	13.7137	14.9572
25	Sharpening filter (0.3)	14.2525	15.5074
26	Sharpening filter (0.5)	14.6631	15.9258
27	Sharpening filter (1)	15.3115	16.5828
28	Laplacian filter (0.2)	5.7964	5.7773
29	Laplacian filter (0.5)	5.7998	5.7649
30	Laplacian filter (0.7)	5.7978	5.7569
31	Laplacian filter (1)	5.7937	5.7472
32	Laplacian of Gaussian filter (5, 0.5)	5.6298	5.6951
33	Laplacian of Gaussian filter (7, 0.5)	5.6099	5.6813
34	Laplacian of Gaussian filter (10, 0.5)	5.6236	5.6577
35	Laplacian of Gaussian filter (5, 0.7)	5.6939	5.6554

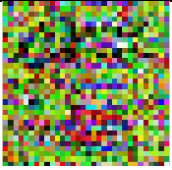

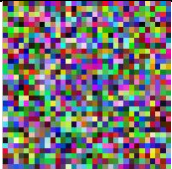

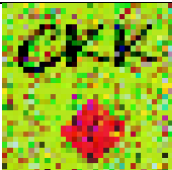

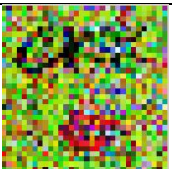



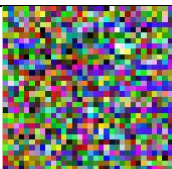







36	Laplacian of Gaussian filter (5, 1.0)	5.4761	5.4875
37	Motion filter (9, 45)	19.5911	20.9359
38	Motion filter (9, 90)	19.448	20.722
39	JPEG compression (45)	24.0993	26.4456
40	JPEG compression (90)	27.9951	37.0079
41	JPEG compression (95)	27.7342	38.2416
42	Bilinear rotation (0.1)	25.4183	26.8419
43	Bilinear rotation (0.2)	20.0218	21.4115
44	Bilinear rotation (0.3)	17.7386	19.0996
45	Bilinear rotation (0.4)	17.0701	18.4092
46	Bicubic rotation (0.1)	26.145	27.5158
47	Bicubic rotation (0.2)	19.4229	20.807
48	Bicubic rotation (0.3)	17.1531	18.5361
49	Nearest rotation (0.1)	Inf	Inf
50	Nearest rotation (0.2)	17.5736	18.9195
51	Nearest rotation (0.3)	16.6069	18.0065
52	Image crop (25)	8.1165	8.2046
53	Image crop (50)	8.9659	9.1082
54	Image crop (75)	11.3732	11.6498
55	Print screen	Inf	Inf
56	Rotation (45)	7.3767	7.4631
57	Rotation (90)	10.1217	10.3945
58	Self similarities (hsv, 001, s, 60)	20.9395	21.8933
59	Self similarities (rgb, 001, s, 60)	22.9736	23.5844
60	Self similarities (yuv, 100, s, 60)	20.3177	21.185



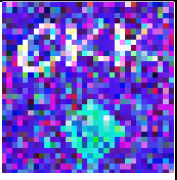

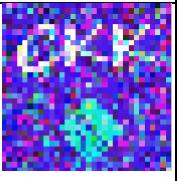

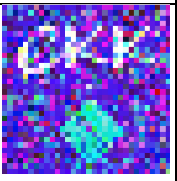

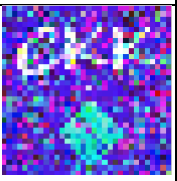

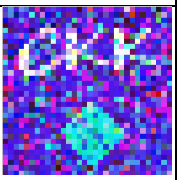

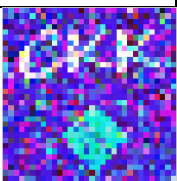

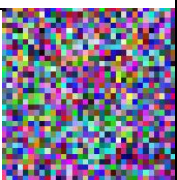

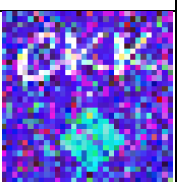



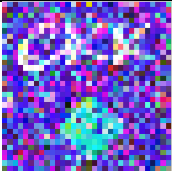

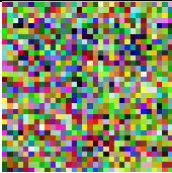

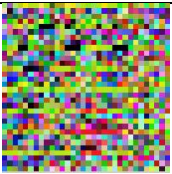

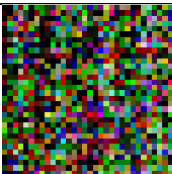

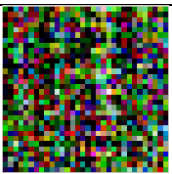

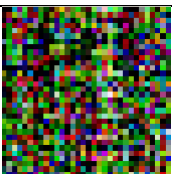

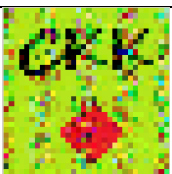

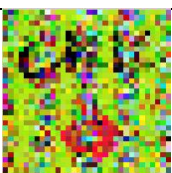

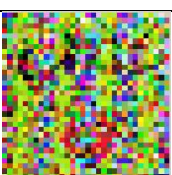

**Table 4.9 NCC comparison for Baboon and CKK watermarked image**





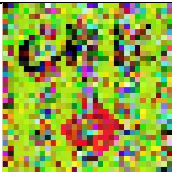







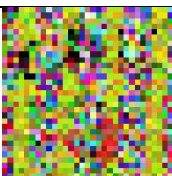





No.	Attack	DCT		Spatial	
			NCC		NCC
1	Gaussian noise (0, 0.001)		0.93619		1
2	Gaussian noise (0, 0.003)		0.8406		0.984742
3	Gaussian noise (0, 0.005)		0.75419		0.955399
4	Gaussian noise (0, 0.01)		0.66618		0.899061
5	Gaussian noise (0, 0.03)		0.49209		0.779343
6	Salt & pepper noise (0.01)		0.88187		1
7	Salt & pepper noise (0.05)		0.59663		1
8	Salt & pepper noise (0.1)		0.46956		1







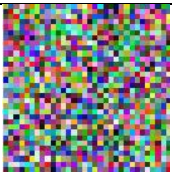

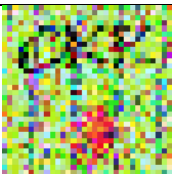



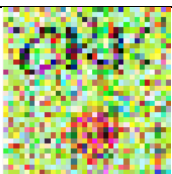

9	Speckle noise (0.01)		0.82545		1
10	Speckle noise (0.05)		0.61035		0.950704
11	Speckle noise (0.1)		0.50763		0.869718
12	Speckle noise (0.5)		0.20401		0.719484
13	Gaussian filter (3, 1)		0.78314		0.901408
14	Gaussian filter (3, 2)		0.54036		0.78169
15	Gaussian filter (3, 3)		0.4897		0.757042
16	Gaussian filter (4, 1)		0.48798		0.746479
17	Gaussian filter (4, 2)		0.12412		0.647887

18	Average filter (3)		0.44057		0.741784
19	Average filter (4)		-0.025541		0.623239
20	Circular average filter (1)		0.90884		0.983568
21	Circular average filter (1.5)		0.637		0.811033
22	Circular average filter (1.7)		0.46029		0.739437
23	Circular average filter (2)		0.030662		0.671362
24	Sharpening filter (0.1)		0.91889		1
25	Sharpening filter (0.3)		0.93101		1
26	Sharpening filter (0.5)		0.93539		1

27	Sharpening filter (1)		0.93834		1
28	Laplacian filter (0.2)		-0.85645		0.517606
29	Laplacian filter (0.5)		-0.86188		0.517606
30	Laplacian filter (0.7)		-0.86298		0.517606
31	Laplacian filter (1)		-0.85648		0.517606
32	Laplacian of Gaussian filter (5, 0.5)		-0.85932		0.517606
33	Laplacian of Gaussian filter (7, 0.5)		-0.85554		0.517606
34	Laplacian of Gaussian filter (10, 0.5)		-0.25103		0.517606
35	Laplacian of Gaussian filter (5, 0.7)		-0.87081		0.517606

36	Laplacian of Gaussian filter (5, 1.0)		-0.77559		0.517606
37	Motion filter (9, 45)		0.29244		0.649061
38	Motion filter (9, 90)		0.27913		0.67723
39	JPEG compression (45)		0.23857		0.678404
40	JPEG compression (90)		0.23048		0.798122
41	JPEG compression (95)		0.24797		0.706573
42	Bilinear rotation (0.1)		0.92482		1
43	Bilinear rotation (0.2)		0.71693		1
44	Bilinear rotation (0.3)		0.46544		0.995305

45	Bilinear rotation (0.4)		0.24586		0.965962
46	Bicubic rotation (0.1)		0.92554		1
47	Bicubic rotation (0.2)		0.71808		1
48	Bicubic rotation (0.3)		0.49317		1
49	Nearest rotation (0.1)		0.99009		1
50	Nearest rotation (0.2)		0.71402		1
51	Nearest rotation (0.3)		0.49317		1
52	Image crop (25)		0.037575		0.53169
53	Image crop (50)		-0.026076		0.523474

54	Image crop (75)		-0.027539		0.524648
55	Print Screen		0.99009		1
56	Rotation (45)		0.010214		0.519953
57	Rotation (90)		-0.0022129		0.526995
58	Self similarities (hsv, 001, s, 60)		0.63806		0.482394
59	Self similarities (rgb, 001, s, 60)		0.9787		0.484742
60	Self similarities (yuv, 100, s, 60)		0.62915		0.485915



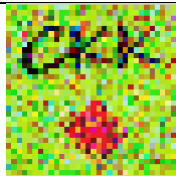



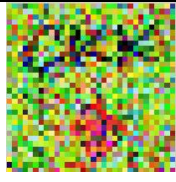

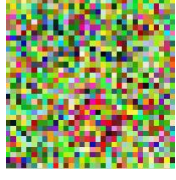





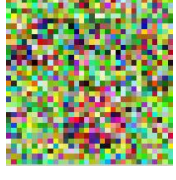



**Table 4.10 PSNR comparison for Pepper and CKK watermarked image**

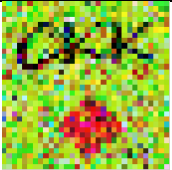

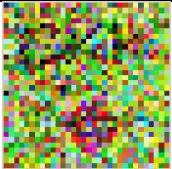




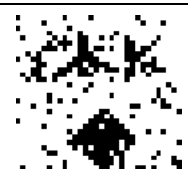






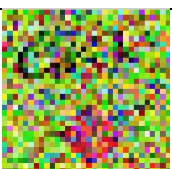

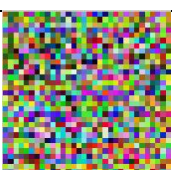

No.	Attack	PSNR	
		DCT	Spatial
1	Gaussian noise (0, 0.001)	30.1252	30.1303
2	Gaussian noise (0, 0.003)	25.4055	25.3913
3	Gaussian noise (0, 0.005)	23.2336	23.2183
4	Gaussian noise (0, 0.01)	20.347	20.313
5	Gaussian noise (0, 0.03)	15.9825	15.9524
6	Salt & pepper noise (0.01)	24.9458	25.0179
7	Salt & pepper noise (0.05)	17.8938	17.9405
8	Salt & pepper noise (0.1)	14.8995	14.925
9	Speckle noise (0.01)	25.9329	25.9315
10	Speckle noise (0.05)	19.2572	19.2179
11	Speckle noise (0.1)	16.5604	16.5598
12	Speckle noise (0.5)	10.7563	10.7839
13	Gaussian filter (3, 1)	26.2111	30.5958
14	Gaussian filter (3, 2)	25.2018	29.6172
15	Gaussian filter (3, 3)	25.0431	29.4632
16	Gaussian filter (4, 1)	24.2295	27.7976
17	Gaussian filter (4, 2)	23.5408	27.1954
18	Average filter (3)	24.9212	29.3448
19	Average filter (4)	23.2843	26.9493
20	Circular average filter (1)	28.3487	32.9804
21	Circular average filter (1.5)	25.8996	30.8311
22	Circular average filter (1.7)	25.2254	30.3514
23	Circular average filter (2)	24.4616	29.5981
24	Sharpening filter (0.1)	15.7722	19.6218
25	Sharpening filter (0.3)	16.6545	21.1054
26	Sharpening filter (0.5)	17.3159	22.217
27	Sharpening filter (1)	18.3026	23.7056
28	Laplacian filter (0.2)	6.4947	6.3005
29	Laplacian filter (0.5)	6.4344	6.2024
30	Laplacian filter (0.7)	6.4087	6.1642
31	Laplacian filter (1)	6.3843	6.1345
32	Laplacian of Gaussian filter (5, 0.5)	6.5638	6.3634
33	Laplacian of Gaussian filter (7, 0.5)	6.5397	6.3502
34	Laplacian of Gaussian filter (10, 0.5)	6.1329	6.1443
35	Laplacian of Gaussian filter (5, 0.7)	6.2396	6.0725

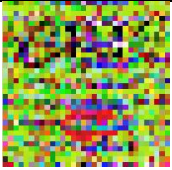

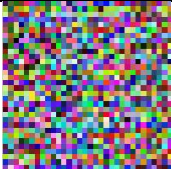





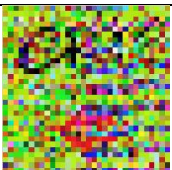

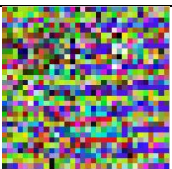











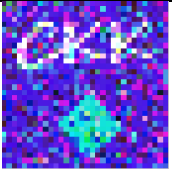

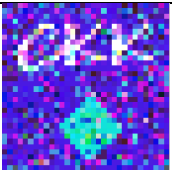

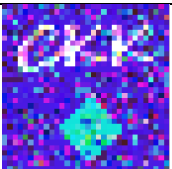

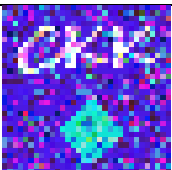

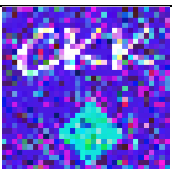

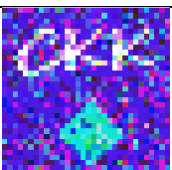

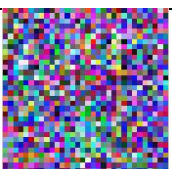

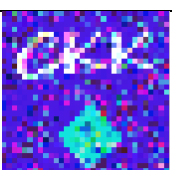

36	Laplacian of Gaussian filter (5, 1.0)	6.0218	5.9836
37	Motion filter (9, 45)	23.7556	27.0396
38	Motion filter (9, 90)	24.0493	27.4863
39	JPEG compression (45)	24.7236	28.6661
40	JPEG compression (90)	26.3032	30.7632
41	JPEG compression (95)	26.6758	31.4726
42	Bilinear rotation (0.1)	30.1044	33.9278
43	Bilinear rotation (0.2)	24.7454	28.481
44	Bilinear rotation (0.3)	22.3808	25.883
45	Bilinear rotation (0.4)	21.4127	24.538
46	Bicubic rotation (0.1)	31.0211	34.8114
47	Bicubic rotation (0.2)	24.2834	28.1374
48	Bicubic rotation (0.3)	21.8871	25.5628
49	Nearest rotation (0.1)	Inf	Inf
50	Nearest rotation (0.2)	22.5068	26.2924
51	Nearest rotation (0.3)	21.2157	24.8842
52	Image crop (25)	10.1445	10.296
53	Image crop (50)	9.8748	10.0296
54	Image crop (75)	10.8279	10.9851
55	Print screen	Inf	Inf
56	Rotation (45)	7.4001	7.4637
57	Rotation (90)	9.6679	9.8395
58	Self similarities (hsv, 001, s, 60)	23.7722	25.338
59	Self similarities (rgb, 001, s, 60)	24.6603	25.3041
60	Self similarities (yuv, 100, s, 60)	23.4792	24.7565

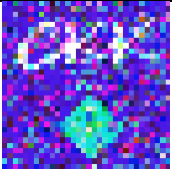

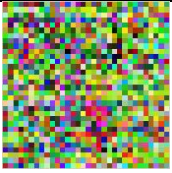

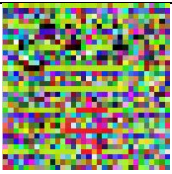

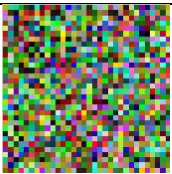

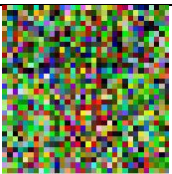

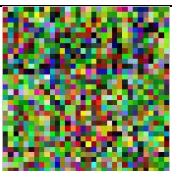



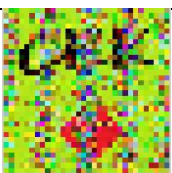

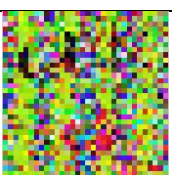

**Table 4.11 NCC comparison for Pepper and CKK watermarked image**

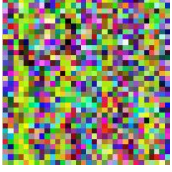



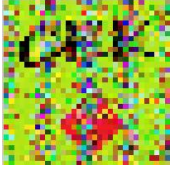

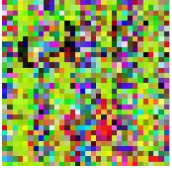





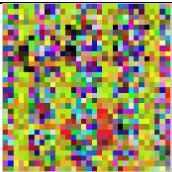





No.	Attack	DCT		Spatial	
			NCC		NCC
1	Gaussian noise (0, 0.001)		0.95651		1
2	Gaussian noise (0, 0.003)		0.84444		0.987089
3	Gaussian noise (0, 0.005)		0.78877		0.957746
4	Gaussian noise (0, 0.01)		0.66576		0.909624
5	Gaussian noise (0, 0.03)		0.46364		0.769953
6	Salt & pepper noise (0.01)		0.86614		1
7	Salt & pepper noise (0.05)		0.60455		1
8	Salt & pepper noise (0.1)		0.43536		1















9	Speckle noise (0.01)		0.83574		1
10	Speckle noise (0.05)		0.58221		0.994131
11	Speckle noise (0.1)		0.42838		0.982394
12	Speckle noise (0.5)		0.19298		0.89554
13	Gaussian filter (3, 1)		0.88121		0.842723
14	Gaussian filter (3, 2)		0.66894		0.725352
15	Gaussian filter (3, 3)		0.61221		0.70892
16	Gaussian filter (4, 1)		0.62247		0.720657
17	Gaussian filter (4, 2)		0.15593		0.629108

18	Average filter (3)		0.56103		0.699531
19	Average filter (4)		-0.058329		0.592723
20	Circular average filter (1)		0.96194		0.962441
21	Circular average filter (1.5)		0.75793		0.742958
22	Circular average filter (1.7)		0.58247		0.683099
23	Circular average filter (2)		0.030774		0.642019
24	Sharpening filter (0.1)		0.9539		1
25	Sharpening filter (0.3)		0.96558		1
26	Sharpening filter (0.5)		0.97185		1

27	Sharpening filter (1)		0.9792		1
28	Laplacian filter (0.2)		-0.88571		0.517606
29	Laplacian filter (0.5)		-0.90006		0.517606
30	Laplacian filter (0.7)		-0.90397		0.517606
31	Laplacian filter (1)		-0.9008		0.517606
32	Laplacian of Gaussian filter (5, 0.5)		-0.89957		0.517606
33	Laplacian of Gaussian filter (7, 0.5)		-0.89802		0.517606
34	Laplacian of Gaussian filter (10, 0.5)		-0.37963		0.517606
35	Laplacian of Gaussian filter (5, 0.7)		-0.92586		0.517606

36	Laplacian of Gaussian filter (5, 1.0)		-0.87118		0.517606
37	Motion filter (9, 45)		0.38062		0.647887
38	Motion filter (9, 90)		0.30709		0.663146
39	JPEG compression (45)		0.24075		0.545775
40	JPEG compression (90)		0.35353		0.683099
41	JPEG compression (95)		0.38224		0.780516
42	Bilinear rotation (0.1)		0.96691		1
43	Bilinear rotation (0.2)		0.76586		1
44	Bilinear rotation (0.3)		0.506		0.997653

45	Bilinear rotation (0.4)		0.22737		0.997653
46	Bicubic rotation (0.1)		0.96678		1
47	Bicubic rotation (0.2)		0.7694		1
48	Bicubic rotation (0.3)		0.50307		1
49	Nearest rotation (0.1)		1		1
50	Nearest rotation (0.2)		0.70278		1
51	Nearest rotation (0.3)		0.4449		1
52	Image crop (25)		-0.015558		0.50939
53	Image crop (50)		0.04841		0.502347

54	Image crop (75)		-0.025082		0.508216
55	Print Screen		1		1
56	Rotation (45)		-0.026515		0.518779
57	Rotation (90)		0.028369		0.507042
58	Self similarities (hsv, 001, s, 60)		0.64616		0.482394
59	Self similarities (rgb, 001, s, 60)		1		0.482394
60	Self similarities (yuv, 100, s, 60)		0.72195		0.482394



















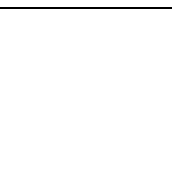

### 4.4.3 Simulation Set 3 Result







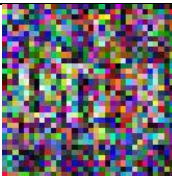

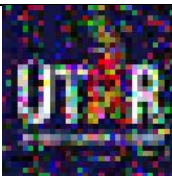

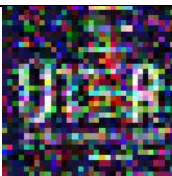

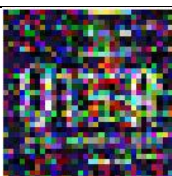





**Table 4.12 PSNR comparison for Lena and UTAR watermarked images**

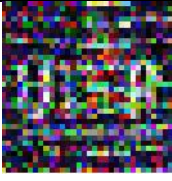





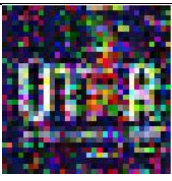

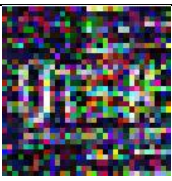

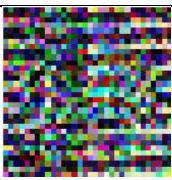







No.	Attack	PSNR	
		DCT	Spatial
1	Gaussian noise (0, 0.001)	30.604	30.6343
2	Gaussian noise (0, 0.003)	25.9934	26.0375
3	Gaussian noise (0, 0.005)	23.9008	23.9296
4	Gaussian noise (0, 0.01)	21.0467	21.0884
5	Gaussian noise (0, 0.03)	16.6751	16.6871
6	Salt & pepper noise (0.01)	24.3707	24.1758
7	Salt & pepper noise (0.05)	17.1929	17.166
8	Salt & pepper noise (0.1)	14.1809	14.151
9	Speckle noise (0.01)	28.7697	28.7944
10	Speckle noise (0.05)	22.0726	22.0911
11	Speckle noise (0.1)	19.2523	19.2993
12	Speckle noise (0.5)	13.2268	13.2761
13	Gaussian filter (3, 1)	28.097	34.1983
14	Gaussian filter (3, 2)	27.0479	33.1045
15	Gaussian filter (3, 3)	26.8832	32.9277
16	Gaussian filter (4, 1)	25.8992	30.3171
17	Gaussian filter (4, 2)	25.1627	29.6215
18	Average filter (3)	26.756	32.7969
19	Average filter (4)	24.8917	29.3433
20	Circular average filter (1)	30.3881	37.4005
21	Circular average filter (1.5)	27.7887	34.8836
22	Circular average filter (1.7)	26.9718	33.8981
23	Circular average filter (2)	26.065	32.5578
24	Sharpening filter (0.1)	17.6467	24.2926
25	Sharpening filter (0.3)	18.4092	25.185
26	Sharpening filter (0.5)	18.9881	25.8333
27	Sharpening filter (1)	19.9194	26.7855
28	Laplacian filter (0.2)	8.9213	8.7101
29	Laplacian filter (0.5)	8.8984	8.6876
30	Laplacian filter (0.7)	8.8857	8.6784
31	Laplacian filter (1)	8.8706	8.6682
32	Laplacian of Gaussian filter (5, 0.5)	8.8809	8.7733



















33	Laplacian of Gaussian filter (7, 0.5)	8.8587	8.7642
34	Laplacian of Gaussian filter (10, 0.5)	8.7244	8.7503
35	Laplacian of Gaussian filter (5, 0.7)	8.7588	8.6185
36	Laplacian of Gaussian filter (5, 1.0)	8.5716	8.55
37	Motion filter (9, 45)	25.3395	29.2358
38	Motion filter (9, 90)	25.4351	29.4314
39	JPEG compression (45)	27.3602	32.3395
40	JPEG compression (90)	30.1668	36.2187
41	JPEG compression (95)	30.6918	37.7533
42	Bilinear rotation (0.1)	32.1416	37.5146
43	Bilinear rotation (0.2)	26.7144	31.8482
44	Bilinear rotation (0.3)	24.2275	28.8575
45	Bilinear rotation (0.4)	23.149	27.0696
46	Bicubic rotation (0.1)	33.0301	38.1899
47	Bicubic rotation (0.2)	26.2739	31.6235
48	Bicubic rotation (0.3)	23.7381	28.5978
49	Nearest rotation (0.1)	Inf	Inf
50	Nearest rotation (0.2)	24.4855	29.8603
51	Nearest rotation (0.3)	22.9421	27.5609
52	Image crop (25)	10.0446	10.1131
53	Image crop (50)	10.4486	10.5203
54	Image crop (75)	11.9017	12.0139
55	Print screen	Inf	Inf
56	Rotation (45)	9.4264	9.4663
57	Rotation (90)	9.9872	10.062
58	Self similarities (hsv, 001, s, 60)	24.1353	25.6719
59	Self similarities (rgb, 001, s, 60)	25.2465	25.6892
60	Self similarities (yuv, 100, s, 60)	23.9229	25.3927









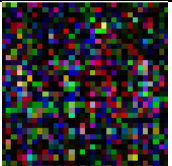

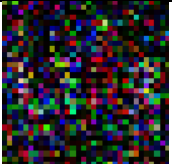







**Table 4.13 NCC comparison for Lena and UTAR watermarked image**




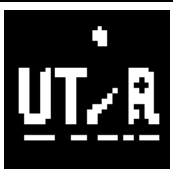














No	Attack	DCT		Spatial	
			NCC		NCC
1	Gaussian noise (0, 0.001)		0.92893		0.9875
2	Gaussian noise (0, 0.003)		0.84013		0.95625
3	Gaussian noise (0, 0.005)		0.78392		0.88125
4	Gaussian noise (0, 0.01)		0.65809		0.85
5	Gaussian noise (0, 0.03)		0.49284		0.775
6	Salt & pepper noise (0.01)		0.86222		1
7	Salt & pepper noise (0.05)		0.54579		1
8	Salt & pepper noise (0.1)		0.481		1

9	Speckle noise (0.01)		0.92222		1
10	Speckle noise (0.05)		0.77788		1
11	Speckle noise (0.1)		0.70044		1
12	Speckle noise (0.5)		0.46673		1
13	Gaussian filter (3, 1)		0.87072		0.59375
14	Gaussian filter (3, 2)		0.66115		0.5625
15	Gaussian filter (3, 3)		0.59292		0.55625
16	Gaussian filter (4, 1)		0.53449		0.5625
17	Gaussian filter (4, 2)		0.10362		0.54375















18	Average filter (3)		0.54427		0.55625
19	Average filter (4)		-0.067787		0.55
20	Circular average filter (1)		0.95219		0.66875
21	Circular average filter (1.5)		0.75837		0.56875
22	Circular average filter (1.7)		0.56324		0.56875
23	Circular average filter (2)		0.045186		0.5625
24	Sharpening filter (0.1)		0.94921		1
25	Sharpening filter (0.3)		0.95423		1
26	Sharpening filter (0.5)		0.95675		1

27	Sharpening filter (1)		0.95948		1
28	Laplacian filter (0.2)		-0.89681		0
29	Laplacian filter (0.5)		-0.90367		0
30	Laplacian filter (0.7)		-0.89664		0
31	Laplacian filter (1)		-0.89693		0
32	Laplacian of Gaussian filter (5, 0.5)		-0.9033		0
33	Laplacian of Gaussian filter (7, 0.5)		-0.89899		0
34	Laplacian of Gaussian filter (10, 0.5)		-0.35223		0.49375
35	Laplacian of Gaussian filter (5, 0.7)		-0.91119		0

36	Laplacian of Gaussian filter (5, 1.0)		-0.82721		0.41875
37	Motion filter (9, 45)		0.29981		0.575
38	Motion filter (9, 90)		0.31798		0.55625
39	JPEG compression (45)		0.27686		0.54375
40	JPEG compression (90)		0.41018		0.5875
41	JPEG compression (95)		0.44703		0.63125
42	Bilinear rotation (0.1)		0.95089		1
43	Bilinear rotation (0.2)		0.75491		0.95625
44	Bilinear rotation (0.3)		0.50019		0.81875

45	Bilinear rotation (0.4)		0.21849		0.725
46	Bicubic rotation (0.1)		0.95173		1
47	Bicubic rotation (0.2)		0.74862		1
48	Bicubic rotation (0.3)		0.49604		0.9625
49	Nearest rotation (0.1)		0.9893		1
50	Nearest rotation (0.2)		0.68055		1
51	Nearest rotation (0.3)		0.43352		1
52	Image crop (25)		-0.024064		0.55
53	Image crop (50)		0.040721		0.5375











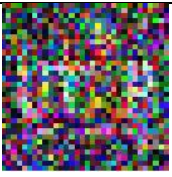



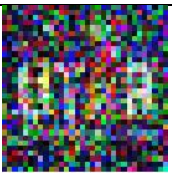
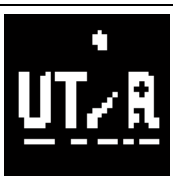


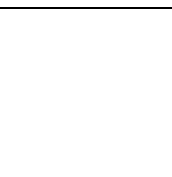

54	Image crop (75)		0.042911		0.53125
55	Print Screen		0.9893		1
56	Rotation (45)		0.0096126		0.56875
57	Rotation (90)		0.019914		0.5625
58	Self similarities (hsv, 001, s, 60)		0.62847		0.50625
59	Self similarities (rgb, 001, s, 60)		0.98938		0.50625
60	Self similarities (yuv, 100, s, 60)		0.80255		0.50625















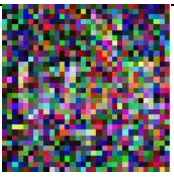

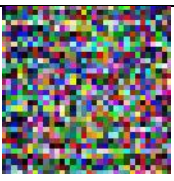

**Table 4.14 PSNR comparison for Baboon and UTAR watermarked images**

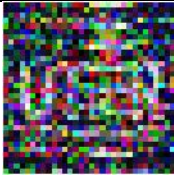





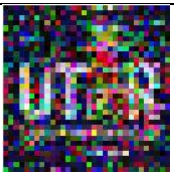

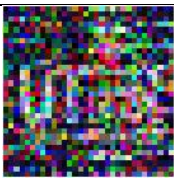

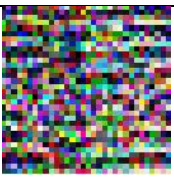


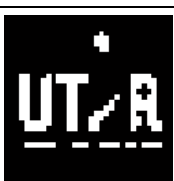




No.	Attack	PSNR	
		DCT	Spatial
1	Gaussian noise (0, 0.001)	30.0515	30.0173
2	Gaussian noise (0, 0.003)	25.3012	25.2945
3	Gaussian noise (0, 0.005)	23.1317	23.1026
4	Gaussian noise (0, 0.01)	20.1968	20.1728
5	Gaussian noise (0, 0.03)	15.7262	15.6942
6	Salt & pepper noise (0.01)	25.2662	25.3252
7	Salt & pepper noise (0.05)	18.1929	18.2593
8	Salt & pepper noise (0.1)	15.2484	15.2518
9	Speckle noise (0.01)	25.513	25.5282
10	Speckle noise (0.05)	18.8186	18.8254
11	Speckle noise (0.1)	16.0219	16.0148
12	Speckle noise (0.5)	10.035	10.0369
13	Gaussian filter (3, 1)	22.6248	24.3375
14	Gaussian filter (3, 2)	21.5942	23.3008
15	Gaussian filter (3, 3)	21.4322	23.1377
16	Gaussian filter (4, 1)	20.9077	22.5147
17	Gaussian filter (4, 2)	20.1472	21.7519
18	Average filter (3)	21.3082	23.0127
19	Average filter (4)	19.8661	21.4629
20	Circular average filter (1)	24.6834	26.4356
21	Circular average filter (1.5)	22.1054	23.8571
22	Circular average filter (1.7)	21.3913	23.1757
23	Circular average filter (2)	20.609	22.3984
24	Sharpening filter (0.1)	13.7674	14.9639
25	Sharpening filter (0.3)	14.3109	15.5128
26	Sharpening filter (0.5)	14.725	15.9298
27	Sharpening filter (1)	15.3782	16.5841
28	Laplacian filter (0.2)	5.8026	5.7718
29	Laplacian filter (0.5)	5.8039	5.7595
30	Laplacian filter (0.7)	5.8009	5.7516
31	Laplacian filter (1)	5.7959	5.7421
32	Laplacian of Gaussian filter (5, 0.5)	5.6428	5.6903
33	Laplacian of Gaussian filter (7, 0.5)	5.6224	5.6768
34	Laplacian of Gaussian filter (10, 0.5)	5.6256	5.6519
35	Laplacian of Gaussian filter (5, 0.7)	5.694	5.6499



















36	Laplacian of Gaussian filter (5, 1.0)	5.4769	5.4819
37	Motion filter (9, 45)	19.6884	20.9376
38	Motion filter (9, 90)	19.5369	20.7261
39	JPEG compression (45)	24.4776	26.4509
40	JPEG compression (90)	28.5333	37.0126
41	JPEG compression (95)	28.846	38.2397
42	Bilinear rotation (0.1)	25.5333	26.8462
43	Bilinear rotation (0.2)	20.1345	21.4157
44	Bilinear rotation (0.3)	17.8403	19.1033
45	Bilinear rotation (0.4)	17.1689	18.4122
46	Bicubic rotation (0.1)	26.2632	27.5215
47	Bicubic rotation (0.2)	19.5368	20.8115
48	Bicubic rotation (0.3)	17.2588	18.54
49	Nearest rotation (0.1)	Inf	Inf
50	Nearest rotation (0.2)	17.6767	18.9247
51	Nearest rotation (0.3)	16.7098	18.0097
52	Image crop (25)	8.1338	8.2047
53	Image crop (50)	8.9554	9.106
54	Image crop (75)	11.3087	11.6484
55	Print screen	Inf	Inf
56	Rotation (45)	7.3764	7.459
57	Rotation (90)	10.0068	10.3948
58	Self similarities (hsv, 001, s, 60)	20.7878	21.8959
59	Self similarities (rgb, 001, s, 60)	22.2441	23.5846
60	Self similarities (yuv, 100, s, 60)	20.0679	21.1871

**Table 4.15 NCC comparison for Baboon and UTAR watermarked images**

No.	Attack	DCT		Spatial	
			NCC		NCC
1	Gaussian noise (0, 0.001)		0.99684		1
2	Gaussian noise (0, 0.003)		0.85056		0.9875
3	Gaussian noise (0, 0.005)		0.76942		0.96875
4	Gaussian noise (0, 0.01)		0.6958		0.91875
5	Gaussian noise (0, 0.03)		0.50917		0.76875
6	Salt & pepper noise (0.01)		0.87976		1
7	Salt & pepper noise (0.05)		0.62413		1
8	Salt & pepper noise (0.1)		0.51188		1










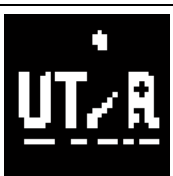








9	Speckle noise (0.01)		0.86168		1
10	Speckle noise (0.05)		0.63746		0.95625
11	Speckle noise (0.1)		0.53776		0.875
12	Speckle noise (0.5)		0.29947		0.78125
13	Gaussian filter (3, 1)		0.81902		0.88125
14	Gaussian filter (3, 2)		0.60925		0.71875
15	Gaussian filter (3, 3)		0.54669		0.70625
16	Gaussian filter (4, 1)		0.48876		0.69375
17	Gaussian filter (4, 2)		0.11369		0.63125









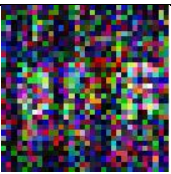

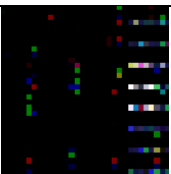



18	Average filter (3)		0.50928		0.7
19	Average filter (4)		-0.048181		0.59375
20	Circular average filter (1)		0.93737		0.9875
21	Circular average filter (1.5)		0.71277		0.7375
22	Circular average filter (1.7)		0.53273		0.7125
23	Circular average filter (2)		0.055307		0.61875
24	Sharpening filter (0.1)		0.93571		1
25	Sharpening filter (0.3)		0.95009		1
26	Sharpening filter (0.5)		0.95565		1

27	Sharpening filter (1)		0.9591		1
28	Laplacian filter (0.2)		-0.86492		0.49375
29	Laplacian filter (0.5)		-0.87396		0.49375
30	Laplacian filter (0.7)		-0.87521		0.49375
31	Laplacian filter (1)		-0.8695		0.49375
32	Laplacian of Gaussian filter (5, 0.5)		-0.88528		0.49375
33	Laplacian of Gaussian filter (7, 0.5)		-0.87812		0.49375
34	Laplacian of Gaussian filter (10, 0.5)		-0.27822		0.49375
35	Laplacian of Gaussian filter (5, 0.7)		-0.88982		0.49375

36	Laplacian of Gaussian filter (5, 1.0)		-0.81133		0.49375
37	Motion filter (9, 45)		0.2986		0.6625
38	Motion filter (9, 90)		0.34539		0.7
39	JPEG compression (45)		0.29889		0.6375
40	JPEG compression (90)		0.29492		0.75625
41	JPEG compression (95)		0.31063		0.69375
42	Bilinear rotation (0.1)		0.92323		1
43	Bilinear rotation (0.2)		0.68974		1
44	Bilinear rotation (0.3)		0.43328		1



45	Bilinear rotation (0.4)		0.17839		0.9625
46	Bicubic rotation (0.1)		0.92467		1
47	Bicubic rotation (0.2)		0.69362		1
48	Bicubic rotation (0.3)		0.43103		1
49	Nearest rotation (0.1)		0.99684		1
50	Nearest rotation (0.2)		0.67312		1
51	Nearest rotation (0.3)		0.41696		1
52	Image crop (25)		-0.044		0.51875
53	Image crop (50)		-0.060795		0.54375







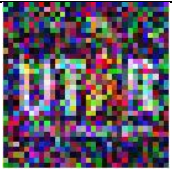











54	Image crop (75)		-0.047573		0.53125
55	Print Screen		0.99684		1
56	Rotation (45)		0.052811		0.49375
57	Rotation (90)		0.019823		0.50625
58	Self similarities (hsv, 001, s, 60)		0.5113		0.50625
59	Self similarities (rgb, 001, s, 60)		0.38005		0.50625
60	Self similarities (yuv, 100, s, 60)		0.47831		0.50625



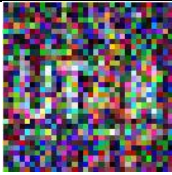

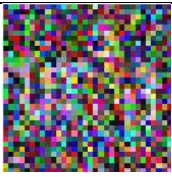

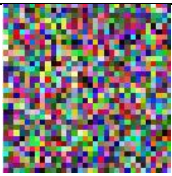











**Table 4.16 PSNR comparison for Pepper and UTAR watermarked images**

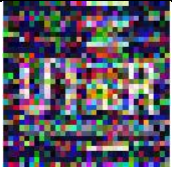







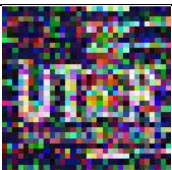









No.	Attack	PSNR	
		DCT	Spatial
1	Gaussian noise (0, 0.001)	30.1049	30.1064
2	Gaussian noise (0, 0.003)	25.4091	25.3913
3	Gaussian noise (0, 0.005)	23.232	23.197
4	Gaussian noise (0, 0.01)	20.3445	20.3154
5	Gaussian noise (0, 0.03)	15.9737	15.9553
6	Salt & pepper noise (0.01)	24.8978	25.0219
7	Salt & pepper noise (0.05)	17.873	17.9432
8	Salt & pepper noise (0.1)	14.8945	14.9281
9	Speckle noise (0.01)	25.9237	25.9274
10	Speckle noise (0.05)	19.2471	19.2146
11	Speckle noise (0.1)	16.5578	16.5564
12	Speckle noise (0.5)	10.7654	10.7801
13	Gaussian filter (3, 1)	26.4304	30.5929
14	Gaussian filter (3, 2)	25.4212	29.6143
15	Gaussian filter (3, 3)	25.2625	29.4604
16	Gaussian filter (4, 1)	24.4129	27.7949
17	Gaussian filter (4, 2)	23.7243	27.1928
18	Average filter (3)	25.1404	29.3416
19	Average filter (4)	23.4668	26.9467
20	Circular average filter (1)	28.5769	32.9784
21	Circular average filter (1.5)	26.1329	30.8278
22	Circular average filter (1.7)	25.4594	30.3477
23	Circular average filter (2)	24.6894	29.5943
24	Sharpening filter (0.1)	15.8888	19.6245
25	Sharpening filter (0.3)	16.789	21.1058
26	Sharpening filter (0.5)	24.4129	22.2144
27	Sharpening filter (1)	18.4654	23.6964
28	Laplacian filter (0.2)	6.4909	6.2968
29	Laplacian filter (0.5)	6.429	6.1988
30	Laplacian filter (0.7)	6.403	6.1607
31	Laplacian filter (1)	6.3784	6.1312
32	Laplacian of Gaussian filter (5, 0.5)	6.5669	6.3594
33	Laplacian of Gaussian filter (7, 0.5)	6.5424	6.3462
34	Laplacian of Gaussian filter (10, 0.5)	6.1336	6.141
35	Laplacian of Gaussian filter (5, 0.7)	6.2349	6.069



















36	Laplacian of Gaussian filter (5, 1.0)	6.0216	5.9801
37	Motion filter (9, 45)	23.9171	27.0375
38	Motion filter (9, 90)	24.2118	27.4845
39	JPEG compression (45)	24.9228	28.676
40	JPEG compression (90)	26.5606	30.7621
41	JPEG compression (95)	26.9528	31.4653
42	Bilinear rotation (0.1)	30.3237	33.9247
43	Bilinear rotation (0.2)	24.9604	28.4799
44	Bilinear rotation (0.3)	22.5703	25.8831
45	Bilinear rotation (0.4)	21.581	24.539
46	Bicubic rotation (0.1)	31.2393	34.807
47	Bicubic rotation (0.2)	24.5048	28.1362
48	Bicubic rotation (0.3)	22.086	25.5638
49	Nearest rotation (0.1)	Inf	Inf
50	Nearest rotation (0.2)	22.7108	26.2909
51	Nearest rotation (0.3)	21.4125	24.8881
52	Image crop (25)	10.1535	10.2999
53	Image crop (50)	9.8887	10.0274
54	Image crop (75)	10.8409	10.9834
55	Print screen	Inf	Inf
56	Rotation (45)	7.4046	7.461
57	Rotation (90)	9.6819	9.8393
58	Self similarities (hsv, 001, s, 60)	23.8632	25.3371
59	Self similarities (rgb, 001, s, 60)	24.659	25.3051
60	Self similarities (yuv, 100, s, 60)	23.496	24.7526

**Table 4.17 NCC comparison for Pepper and UTAR watermarked images**



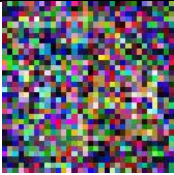



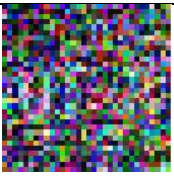

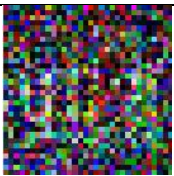

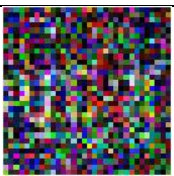







No.	Attack	DCT		Spatial	
			NCC		NCC
1	Gaussian noise (0, 0.001)		0.95264		1
2	Gaussian noise (0, 0.003)		0.86736		0.99375
3	Gaussian noise (0, 0.005)		0.8057		0.975
4	Gaussian noise (0, 0.01)		0.68296		0.9375
5	Gaussian noise (0, 0.03)		0.46705		0.7875
6	Salt & pepper noise (0.01)		0.8932		1
7	Salt & pepper noise (0.05)		0.59868		1
8	Salt & pepper noise (0.1)		0.45446		1







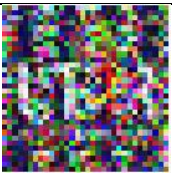





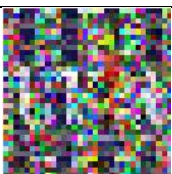





9	Speckle noise (0.01)		0.83475		1
10	Speckle noise (0.05)		0.58192		0.99375
11	Speckle noise (0.1)		0.41252		0.99375
12	Speckle noise (0.5)		0.19241		0.925
13	Gaussian filter (3, 1)		0.8883		0.83125
14	Gaussian filter (3, 2)		0.70054		0.75
15	Gaussian filter (3, 3)		0.65518		0.7375
16	Gaussian filter (4, 1)		0.59965		0.70625
17	Gaussian filter (4, 2)		0.18874		0.65625





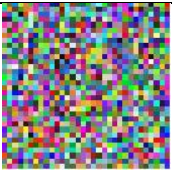









18	Average filter (3)		0.59713		0.725
19	Average filter (4)		-0.020288		0.65
20	Circular average filter (1)		0.96449		0.9625
21	Circular average filter (1.5)		0.77367		0.75625
22	Circular average filter (1.7)		0.60741		0.69375
23	Circular average filter (2)		0.082901		0.61875
24	Sharpening filter (0.1)		0.97411		1
25	Sharpening filter (0.3)		0.98337		1
26	Sharpening filter (0.5)		0.98957		1

27	Sharpening filter (1)		0.98727		1
28	Laplacian filter (0.2)		-0.90125		0.49375
29	Laplacian filter (0.5)		-0.9113		0.49375
30	Laplacian filter (0.7)		-0.90965		0.49375
31	Laplacian filter (1)		-0.90588		0.49375
32	Laplacian of Gaussian filter (5, 0.5)		-0.92053		0.49375
33	Laplacian of Gaussian filter (7, 0.5)		-0.91406		0.49375
34	Laplacian of Gaussian filter (10, 0.5)		-0.40472		0.49375
35	Laplacian of Gaussian filter (5, 0.7)		-0.93087		0.49375



36	Laplacian of Gaussian filter (5, 1.0)		-0.8809		0.49375
37	Motion filter (9, 45)		0.37352		0.625
38	Motion filter (9, 90)		0.31361		0.65625
39	JPEG compression (45)		0.37972		0.55625
40	JPEG compression (90)		0.40554		0.68125
41	JPEG compression (95)		0.43607		0.775
42	Bilinear rotation (0.1)		0.96704		1
43	Bilinear rotation (0.2)		0.76684		1
44	Bilinear rotation (0.3)		0.47903		0.9875

45	Bilinear rotation (0.4)		0.20177		0.9625
46	Bicubic rotation (0.1)		0.96952		1
47	Bicubic rotation (0.2)		0.75984		1
48	Bicubic rotation (0.3)		0.48777		1
49	Nearest rotation (0.1)		1		1
50	Nearest rotation (0.2)		0.67343		1
51	Nearest rotation (0.3)		0.41739		1
52	Image crop (25)		0.005525		0.58125
53	Image crop (50)		-0.0041487		0.58125

54	Image crop (75)		0.020835		0.54375
55	Print Screen		1		1
56	Rotation (45)		0.017557		0.49375
57	Rotation (90)		-0.015978		0.5875
58	Self similarities (hsv, 001, s, 60)		0.66798		0.50625
59	Self similarities (rgb, 001, s, 60)		1		0.50625
60	Self similarities (yuv, 100, s, 60)		0.71095		0.50625

## 4.5 Discussion

### 4.5.1 Simulation Set 1

As for simulation set 1, all the watermarks are extracted from host images without any attacks applied on them. Alpha value for DCT algorithm is 15, while spatial domain uses  $\alpha = 5$ . Comparison of SNR, PSNR and NCC values of these two algorithms are shown as above.

SNR values for DCT varies from 18dB to 20dB, whereas spatial domain records the values between 31dB and 33dB. Meanwhile, PSNR for DCT can be averaged to 26dB, and spatial domain sets the average value of 39dB.

By comparing DCT and spatial domain embed results, spatial domain takes the lead as it has higher PSNR value and resembles the original host images. Despite the fact that all three watermarked images have different SNR and PSNR, all of them are almost identical to their host images. In addition, all the extracted watermarks have a satisfying NCC result, which are almost equals to 1.

### 4.5.2 Simulation Set 2 & 3

By judging from PSNR aspect, both DCT-based and spatial-based watermarking can withstand most of the attacks applied on them, except for Laplacian filters, rotations and image cropping as they have the average value less than 15dB. The quality of the images degrades enormously after these attacks are applied on them.

On the contrary, the extracted watermarks and NCC values prove that DCT and spatial-based watermarking performs cogently under particular attacks. For both watermarking algorithms, they can perform justly Gaussian noise, salt & pepper noise, speckle noise, sharpening filter and print screen attack. However, Gaussian filter will render them useless, especially for spatial domain watermarking. In addition, circular average filter of higher radius will cause the NCC to drop dramatically.

Moreover, Laplacian filter caused the images to have very low PSNR values, however this do not affect the extraction process of watermark embedded in it. For DCT-based watermarking, it has NCC value about -1, while spatial-based watermarking reaches  $NCC = 0$ , these will not cause any problem as the extracted watermarks are in their negative region, thus can be easily compared to their respective original watermarks.

Furthermore, when the watermarked images are compressed by JPEG, they return an abominable NCC result, which are lower than 0.4 and 0.6 for DCT-based and spatial-based respectively. Besides, image cropping causes both watermarking algorithms to have relatively bad NCC result.

Contra wise, both algorithms manage to survive through print screen attack, yielding the NCC result approximately 1. Else for self similarities tests, DCT-based watermarking has better result compared to spatial-based, where spatial-based only has NCC less than 0.5.

To sum it all, spatial-based watermarking has better robustness in noise adding, certain types of filtering and rotation, compared to DCT-based. However, DCT-based watermarking has its strength in particular filtering, as well as self similarities tests.

## **Chapter 5 CONCLUSION AND FUTURE WORK**

By comparing DCT-based and spatial-based watermarking algorithm, each of them has its own strength, as well as weaknesses. In terms of imperceptibility, spatial-based algorithm takes the lead as the watermarked images from spatial domain algorithm have higher SNR and PSNR.

Meanwhile, spatial-based watermark is more robust towards noise attacks compare with DCT-based. As for image filtering attack, both algorithms have their equal strength in certain attacks. However, JPEG compression renders both watermark algorithms useless. Besides, both DCT-based and spatial-based are strong against print screen attack. On the other hand, during self similarities tests, DCT-based algorithm has the higher NCC values, which means more robust compare to spatial-based.

Furthermore, judging from speed of embed and extract of each algorithm, DCT-based once again prove its strength by having shorter processing time. Next, for security wise, spatial-based has better security as the watermark will undergo XOR process before being embedded into the host image, thus harder to be traced. In terms of complexity, spatial-based is more complex due to the pre-processing of the host image before embed and extract process, thus causing the time for each process to increase as well.

In the future, this project can be further enhanced by increasing the number of attacks to the watermarked images, besides adding in more watermarking algorithms. More tests can be carried out to determine the robustness of each algorithm from different aspect, while adding in more algorithms can have a better comparison among the various types of watermarking method, hence helping others to choose a better algorithm for implementation. In addition, both DCT-based and spatial-based algorithms can be further improved to obtain better watermarked results, as well as faster respond time.

## REFERENCE

1. Li Xiaoni; Sun Xiaoying; Wang Dazhong, "*Real-Coded Generic Algorithm for Optimized Digital Watermarking Embedding in Time-Domain*", in *The Ninth International Conference on Electronic Measurement & Instruments*. 2009.
2. A. Sverdlov; S. Dexter; A. Eskicioglu. "*Robust DCT-SVD domain image watermarking for copyright protection: Embedding data in all frequencies*". in *International Multimedia Conference*. 2004. Germany,.
3. E. Fu, "*Literature survey on digital image watermarking*", in *EE381K-Multidimensional Signal Processing*. 1998.
4. Snow, C.P., in *New York Times*. 15 March 1971.
5. B. Mohan ; S. Kumar, "*A robust image watermarking scheme using singular value decomposition*". *J. Multimedia*, May 2008. **vol. 3**(no. 1).
6. Sami Baba; Lala Krikor; Thawar Arif; Ziyad Shaaban, "*Watermarking of Digital Images in Frequency Domain*". *International Journal of Automation and Computing*.
7. Miller, M.C., I.J.; Linnartz, J.P.M.G.; Kalker, T, "*A review of watermarking principles and practices*". In *Digital Signal Processing in Multimedia Systems*, 1999: p. 461-485.
8. Schneier, B., "*Applied Cryptography*". 2nd ed. 1996, New York: Wiley.
9. Kahn, D., "*The History of Stagenography*". *Information Hiding*, 1996: p. 1174.
10. Chu, W.C., "*DCT-based image watermarking using subsampling*". *IEEE Trans. Multimedia*, 2003. **5**(1): p. 34-38.
11. M. Barni; F. Bartolini; A. Piva, "*Improved wavelet-based watermarking through pixel-wise masking*". *Image Process. IEEE Trans. Image Process*, 2001. **10**(5): p. 783-791.
12. J.J.K. O'Ruanaidh; W.J. Dowling; F.M. Boland, "*Phase watermarking of digital image, Proceedings of the IEEE International Conference on Image Processing*, 1996. **Vol.3**: p. 239-242.
13. N.Nikolaidis; I.Pitas, "*Robust image watermarking in the spatial domain*". *Signal Process.*, 1998. **66**: p. 385-403.



14. Jo.M.Kim.H, "A digital image watermarking scheme based on vector quantisation". IEICE Trans. Inf.&Syst., 2002. **Vol E85-D**: p. 303-305.
15. Berghel, H., "Digital watermarking makes it mark". netWorker: The craft of network computing, 1998. **2**(4): p. 30-39.
16. Ajit Kulkarni, "Digital Watermarking", Virginia Tech.
17. Cox I. J.; Miller, M.L.B.J.A., "Digital Watermarking". 2002, USA: Morgan Kaufmann Publishers.
18. Katzenbeisser S. ; Petitcolas F. A. P., "Information Hiding Techniques for Steganography and Digital Watermarking". 2000, UK: Artech House.
19. Cox I.; Miller M. ; Bloom J., "Watermarking applications and their properties", in *Proceedings of the international conference on information technology: Coding and computing*. 27-29 March 2000: Las Vegas, Nevada.
20. Collberg C.; Thomborson C., "Software watermarking: Models and dynamic embeddings", in *Proceedings of the 26th ACM SIGPLAN-SIGACT*. 20-22 January 1999: San Antonio, Texas.
21. Fabien A. P. Petitcolas ; Markus G. Kuhn. "StirMark 2". November 1997; Available from: <http://www.cl.cam.ac.uk/~fapp2/watermarking/stirmark/>.
22. Edin Muharemagic ; Borko Furht, "Survey Of Watermarking Techniques And Applications", in *Department of Computer Science and Engineering*, Florida Atlantic University: U.S.A. p. 30.
23. Certimark. "Common data processing and intentional attacks". IST- 1999 - 10987 1999; Available from: <http://www.certimark.org/>.
24. Chiuo-Ting Hsu; Ja-Ling Wu, "Hidden Digital Watermarks in Images". IEEE Transactions On Image Processing, January 1999. **Vol. 8**(No. 1).
25. J.T. Brassil; S. Low; N.F. Maxemchuk; L. O’Gorman, "Electronic marking and identification techniques to discourage document copying". IEEE J. Select. Areas Commum, Oct. 1995. **vol. 13**: p. 1495–1504.
26. I. Pitas;T. H. Kaskalis, "Applying signatures on digital images". Proc. IEEE Nonlinear Signal and Image Processing, June 1995: p. 460–463.
27. O. Bruyndonckx; J. J. Quisquater; B. Macq, "Spatial method for copyright labeling of digital images". Proc. IEEE Nonlinear Signal and Image Processing, June 1995: p. 456–459.
28. Walton, S., "Image authentication for a slippery new age". Dr. Dobb’s J, Apr. 1995: p. 18–26.

29. W. Bender; D. Gruhl; N. Morimoto, "*Techniques for data hiding*". Proc. SPIE, Feb. 1995. **vol. 2420**: p. 40.
30. E. Koch; J. Zhao, "*Toward robust and hidden image copyright labeling*". Proc. IEEE Nonlinear Signal and Image Processing, June 1995: p. 452–455.
31. I. J. Cox; J. Kilian; T. Leighton; T. Shammoon, *Secure spread spectrum watermarking for multimedia*, in *Tech. Rep. 95-10*. 1995., NEC Res. Inst., Princeton, NJ.
32. M. D. Swanson; B. Zhu; A. H. Tewfik, "*Transparent robust image watermarking*". Proc. ICIP'96: p. 211–214.
33. Sklar, B., *Digital Communications*. Englewood Cliffs. 1988: NJ: Prentice Hall.
34. W. B. Pennebaker; J. L. Mitchell, *JPEG: Still Image Data Compression Standard*. New York: Van Nostrand Reinhold, 1993.: p. 34–38.
35. Mei Jiansheng; Li Sukang; Tan Xiaomei, "*A Digital Watermarking Algorithm Based On DCT and DWT*". Proceedings of the 2009 International Symposium on Web Information Systems and Applications, May 2009: p. 104-107.
36. J.C. Yen, "*Watermark Embedded in Electronic Letters*", in *Xi'an:XiDian University Press*. 2000., p. 80-81.
37. Ibrahim Nasir; Ying Weng; Jianmin Jiang, *Novel Multiple Spatial Watermarking Technique in Color Images*. Fifth International Conference on Information Technology: New Generations.

## **APPENDIX A: BIWEEKLY REPORT**

## FINAL YEAR PROJECT BIWEEKLY REPORT (PROJECT 2)

Trimester, Year: T3, Y3	Study week no.: 1
Student Name & ID: Chua Kah Keong 08ACB03342	
Supervisor: Mr. Leong Chun Farn	
Project Title: Development and Analysis of Spatial Domain and Transform Domain Watermarking Technique	

<p><b>1. WORK DONE</b></p> <p>Successfully implemented the embed process of DCT algorithm; will proceed to the extraction process.</p>
<p><b>2. WORK TO BE DONE</b></p> <p>DCT extraction process, attacks on DCT watermarked images. Spatial domain algorithm.</p>
<p><b>3. PROBLEMS ENCOUNTERED</b></p> <p>The quality of DCT embedded images are low in quality, do not meet the requirement set.</p>
<p><b>4. SELF EVALUATION OF THE PROGRESS</b></p> <p>Up to pace, completed the embed algorithm according to the schedule.</p>

\_\_\_\_\_  
Supervisor's signature

\_\_\_\_\_  
Student's signature

## FINAL YEAR PROJECT BIWEEKLY REPORT (PROJECT 2)

Trimester, Year: T3, Y3	Study week no.: 3
Student Name & ID: Chua Kah Keong 08ACB03342	
Supervisor: Mr. Leong Chun Farn	
Project Title: Development and Analysis of Spatial Domain and Transform Domain Watermarking Technique	

<p><b>1. WORK DONE</b></p> <p>Managed to debug the errors in DCT extraction process, hence completed the overall coding for DCT algorithm.</p>
<p><b>2. WORK TO BE DONE</b></p> <p>Perform attacks on DCT watermarked images, and then obtain the PSNR and NCC values. Spatial domain algorithm.</p>
<p><b>3. PROBLEMS ENCOUNTERED</b></p> <p>Encountered errors in DCT extraction process, later on resolve it successfully with the guidance from supervisor, Mr. Leong.</p>
<p><b>4. SELF EVALUATION OF THE PROGRESS</b></p> <p>Ought to red more on MATLAB coding, lack of knowledge in this area. Progress is moderate.</p>

\_\_\_\_\_  
Supervisor's signature

\_\_\_\_\_  
Student's signature

## FINAL YEAR PROJECT BIWEEKLY REPORT (PROJECT 2)

Trimester, Year: T3, Y3	Study week no.: 5
Student Name & ID: Chua Kah Keong 08ACB03342	
Supervisor: Mr. Leong Chun Farn	
Project Title: Development and Analysis of Spatial Domain and Transform Domain Watermarking Technique	

### 1. WORK DONE

Completed the attacks and benchmarking of DCT watermarked images, will proceed to spatial-based algorithm.

### 2. WORK TO BE DONE

Arrange the results of DCT-based algorithm into tables, so can be easily compared with spatial-based algorithm.  
Spatial domain embeds and extracts process.

### 3. PROBLEMS ENCOUNTERED

Ambiguous about host image pre-processing of spatial-based algorithm.  
Will clarify with supervisor.

### 4. SELF EVALUATION OF THE PROGRESS

Progress is good, however still need to speed up as more time is required for attacks and benchmark process.

\_\_\_\_\_  
Supervisor's signature

\_\_\_\_\_  
Student's signature

## FINAL YEAR PROJECT BIWEEKLY REPORT (PROJECT 2)

Trimester, Year: T3, Y3	Study week no.: 7
Student Name & ID: Chua Kah Keong 08ACB03342	
Supervisor: Mr. Leong Chun Farn	
Project Title: Development and Analysis of Spatial Domain and Transform Domain Watermarking Technique	

### 1. WORK DONE

Embed and extract algorithm of spatial-based watermarking are implemented, however fail to function flawlessly.

### 2. WORK TO BE DONE

Further improvement of embed and extract process of spatial-based watermarking.  
Benchmarking of spatial-based watermarking.

### 3. PROBLEMS ENCOUNTERED

Spatial-based watermarking embed process perform capably, however fail to extract watermarked from host images.

### 4. SELF EVALUATION OF THE PROGRESS

Slow, procrastination occurs due to mid-terms and poor time management. Will allocate more time once mid-terms and assignments are completed.

\_\_\_\_\_  
Supervisor's signature

\_\_\_\_\_  
Student's signature

## FINAL YEAR PROJECT BIWEEKLY REPORT (PROJECT 2)

Trimester, Year: T3, Y3	Study week no.: 9
Student Name & ID: Chua Kah Keong 08ACB03342	
Supervisor: Mr. Leong Chun Farn	
Project Title: Development and Analysis of Spatial Domain and Transform Domain Watermarking Technique	

### 1. WORK DONE

After seeking guidance from Mr. Leong, realised that spatial-based algorithm done previously was incorrect, hence redo the implementation.

### 2. WORK TO BE DONE

Embed and extract algorithm of spatial-based watermarking.  
Benchmarking of spatial-based watermarking.  
Comparison between DCT and spatial-based algorithms.

### 3. PROBLEMS ENCOUNTERED

Errors in both spatial-based embed and extraction. Resolved by restarting the implementation process.

### 4. SELF EVALUATION OF THE PROGRESS

Very slow, currently is way behind the schedule planned. Need to expedite the progress.

\_\_\_\_\_  
Supervisor's signature

\_\_\_\_\_  
Student's signature



## FINAL YEAR PROJECT BIWEEKLY REPORT (PROJECT 2)

Trimester, Year: T3, Y3	Study week no.: 11
Student Name & ID: Chua Kah Keong 08ACB03342	
Supervisor: Mr. Leong Chun Farn	
Project Title: Development and Analysis of Spatial Domain and Transform Domain Watermarking Technique	

<p><b>1. WORK DONE</b></p> <p>Completed both the DCT and spatial-based watermarking, summarized the simulated results into tables. Comparisons are made between both algorithms.</p>
<p><b>2. WORK TO BE DONE</b></p> <p>Finalize the report, check for grammatical errors and final touch up.</p>
<p><b>3. PROBLEMS ENCOUNTERED</b></p> <p>Future work of the project.</p>
<p><b>4. SELF EVALUATION OF THE PROGRESS</b></p> <p>Good and satisfied, managed to complete the project by the due date.</p>

\_\_\_\_\_  
Supervisor's signature

\_\_\_\_\_  
Student's signature

## **APPENDIX B: MATLAB CODES**