Robust and Adaptive Watermarking Technique for Secure Authorship of Digital Images

Prepared by

G.W.R. Sandaruwan

149231R

Faculty of Information Technology

University of Moratuwa

May 2017

This thesis submitted in partial fulfillment of the requirement for the degree of MSc in IT of University of Moratuwa

Declaration

I, GALLENA WATTHAGE REKA SANDARUWAN declare that this thesis and the work presented in it are my own and has been generated by me as the result of my own original research. This research work not been submitted for a degree in any other university/institution before.

I confirm that:

1. This work was done wholly or mainly while in candidature for a Master Degree at this University.
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
3. Where I have consulted the published work of others, this is always clearly attributed.
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
5. Either none of this work has been published before submission.

G.W.R. Sandaruwan: …..... Date: / /

Dr. Lochandaka Ranathunga: …..... Date: / /

(Head of Department / Senior Lecturer)

B.Sc. Sp(Hons), M.Sc., PGDip in DEd. (IGNOU),

PhD (Malaya), MIPSL, MCSSL

Dedicated to

Dr. Lochandaka Ranathunga

Acknowledgments

I would like to express my sincere gratitude to My Supervisor Dr. Lochandaka Ranathunga Head of Department of Information Technology & Senior Lecture of University of Moratuwa for the continuous support of my M.Sc. study and related research, for his thought fullness, experience, patience, motivation, and immense knowledge. His guidance helped me in all the time of conduct the research and exposure My knowledge of research aria. I could not have imagined better advisor and mentor rather than Dr. Lochandaka Ranathunga in my whole life.

I would also like to acknowledge the rest of lectures in Faculty of Information Technology: Mr. D.K. Withanage (Former Dean of FIT), Mr. Saminda Premarathna (Senior Lecture), Mr. B.H. Sudantha (Course Coordinator), Mr. C.P. Wijesiriwardhana (Senior Lecture) for their insightful comments and encouragement.

Last but not the least, I would like to thank My colleagues and family: My parents and family members for supporting me spiritually throughout and my life in general.

G.W.R. Sandaruwan.

Abstract

There has been an increase in the broadcasting media since the last thirty years. The Tera Byte level multimedia data has been created, copied and transmitted via Internet every second. The access, sharing, replication and manipulation of images have become daily needs. Originators mind has a fear of illegal distribution and violation of copyright protection by malicious users. Hidden digital watermarking techniques have come to the rescue as a powerful solution to such potential problems. Several hidden type watermarking techniques have been proposed with a variety of their usage, complexity and security which are the primary concerns of such technique.

Digital watermarking describes methods and technologies that embed hidden information in digital media, such as images, video, audio or any other kind of multimedia object. Hidden digital watermarking techniques have many more challenges such as robustness, fidelity, extraction, and capacity. Robustness is a considerable feature of hidden watermarking techniques. This feature refers to the ability to detect the watermark after some signal processing operation, for example rotation, scaling, compression, noise adding, and image cutting. Resistance against several types of attacks is a real challenge for researcher's long years back. The robustness feature of hidden digital watermark will give huge benefits. Fidelity requirement of watermarking could be called invisibility. Fidelity feature preserves the similarity between the watermarked image and the original image according to human perception. The watermark must remain invisible notwithstanding the occurrence of small degradations in image features, contrast and brightness. Meeting the psycho-visual fidelity criteria is very important to every digital watermarking technique. Psycho-visual fidelity, is a huge challenging research gap of digital watermarking. Several previous researches have shown the evidence that invisibility and robustness are both very difficult to maintain mutually in available watermarking methodologies. Adaptive extraction feature determines, which resources are necessary for the analysis to extract the watermark from the watermarked image. Resource requirement should be minimized by a good watermark detection technique. The number of bits that can be inserted through watermark embedding process is a considerable research challenge to hidden digital watermarking. Increasing capacity of watermarking methods and fidelity of watermark can be conflicting. These two goals of watermarking

techniques should be balanced without any conflict.

This thesis describes a novel approach to hidden digital watermarking, based on low level features of digital image. It was hypothesized that, the above problem can be solved by a novel method of invisible watermarking of the digital images, based on low level features of the image and transform domain techniques. Corners are the salient feature of digital image. Corner detection is very important to the image processing operation. The Harris operator has been widely used for corner detection. This thesis has proposed a novel corner detector which is an extended and improved version of Harris operator. Novel operator is a step by step process, which improves corner detection ability and scale invariant property. The novel solution proposed in this thesis will giving the guarantee of features which robustness, fidelity, capacity and adaptive extraction. Proposed method in this thesis is to divide the entire process in to three major parts which are the analysis of the low-level features of image and detecting of the corner points in original or host image, watermark embedding into detected corner points, and adaptive watermark extraction. This study has abstracted a novel model by supporting Sobel operator of edge detection and Laplacian of Gaussian (LoG) filter. Using Sobel operator x-direction, y-direction and diagonal directions over host image, having improved edge detection ability of novel operator. LoG filter has provided a smoothing property and it's less sensitive to noise. LoG filter has improved scale invariant property of novel operator. Proposed watermarking solution can be used on color images and watermark object also can be small color images. Recover data matrix have generated by analyzing and comparing features of host image and intensity values of watermark image. Thus, recover data is generated dynamically. Generated recover data has been embedded in to host image at the prominent corner points of host image. Corner points are immutable points of image against many types of image processing operation. Specially corners provide good surveillance against rotation operations. Novel operator of this thesis has used LoG filter for the purpose of smoothing the host image. Thus, it has provided a good surveillance against scaling operation. Generated recover data are light weight. Due to the recover data generation process, it compares the host image and watermark object. Embedding process achieves the minimum degradation to original image. Thus, the novel approach of watermarking has guaranteed robustness and fidelity characteristics.

After embedding the recover data into host image, it produces uncompressed watermarked image. Study has proposed, represent watermarked image in a more efficient transferable and store-able manner. For the purpose of efficient representation, row-watermarked object has been converted into encoded format. Discrete Cosine Transformation (DCT) has been used to encode the row watermarked object. Other major function of watermarking system is an extracting watermark from watermarked object. Extraction process also has used low level features of watermarked image. Proposed extraction method of thesis is an adaptive process. It required minimum number of inputs included in watermarked image and meta data of watermark only. Extraction process has never required original image.

Comprehensive experiments have been conducted for the testing of novel watermarking approach. Study has used common data set widely used in image processing experiments. Experimental environment was a prototype application developed in C/C++ programing language. Evaluation process has been designed by covering all characteristics of watermarking algorithm. This thesis represents a complete evaluation of novel proposed solution by using a large data set. The thesis has represented test data results and analysis of results according to major characteristics of watermarking systems.

Evaluation has given evidence, that novel feature detection operator and watermark embedding algorithm provide higher robustness against rotation, scaling, filtering and noise adding attacks. The experimental results have given evidence that novel approach of digital watermarking can balance in between robustness and fidelity vis versa. Extraction method proposed by this thesis is a minimum number of inputs and it never required original image. Conclusion is that the proposed approach of digital watermarking gives many advantages over available methods.

Table of contents

List of Tables

List of Figures

Abbreviations

LoG:	Laplacian of Gaussian
DCT:	Discrete Cosine Transformation
I-DCT:	Inverse Discrete Cosine Transformation
DWT:	Discrete Wavelet Transformation
MTWC:	Multi Threshold Wavelet Codec
JPEG:	Joint Photographic Experts Group
SVD:	Singular Value Decomposition
LWT:	Lifting Wavelet Transform
DFRNT:	Discrete Fractional Random Transform
FFT:	Fast Fourier Transform
RT:	Ridgelet Transform
EHD:	Edge Histogram Descriptor
LSB:	Least Significant Bit
ITU:	International Telecommunications Union
GCC:	GNU Compiler Collection
MSE:	Mean Square Error
PSNR:	Peak Signal to Noise Ratio
NCC:	Normalized Cross-Correlation
NAE:	Normalized Absolute Error
SSIM:	Structural Similarity Index Matrix
FSIM:	Feature Similarity Index Matrix
PVD:	Pixel Value Differencing
TPVD:	Try-way Pixel Value Differencing

Chapter 1

Introduction

1.1 Introduction

There has been an increase in the use of broadcasting media since the last thirty years, protecting of the authentication of multimedia contents is a challenging research area nowadays. Reason is that many techniques had been developed to solve this problem. Invisible watermarking seems to be the most popular among many researchers around the world. Digital watermarks can be used for a lot of applications such as copyright protection, owner identification and distribution monitoring over the network.

Increased Internet usage has necessitated a technique that is able to protect the copyright of multimedia content such as digital images and digital videos. The aim of invisible watermark is to improving the authorization and protect the copyrights of real authors.

There is a research gap in this study, areas that inadequate attention has been given to the digital watermarking techniques and to find a solution for secure authorship of digital images. Hypothesis of this study is mentioned problem which can be solved by introducing the novel feature base digital watermark embedding method and adaptive watermark extraction method.

The novel approach proposed in this study, assures the general features of watermarking such as robustness, fidelity, capacity and adaptive extraction. This study can be applied to several areas of applications such as copy control, owner identification, digital signatures and fingerprinting.

1.2 Background & Motivation

As a result of the fast and extensive growth of network technology, digital information can be distributed with no quality loss and low cost delivery. Protection of multimedia content has recently become an important issue because the authors want to claim ownership of their intellectual properties. Thus, over the last several years, digital information science has emerged to seek answers to copyright protection issues. Several researches have been

conducted on the study of protecting the copyright of digital contents storing, transmitting, and processing. Today protecting the ownership of digital images has become a very challenging research area.

Digital watermarking [1], [2] is the process of embedding or hiding digital information into another digital product. After embedding the watermark, then embedded data can later be extracted or detected from the watermarked product. Invisible watermarking of digital image could to be used in many different applications [2], such as ownership evidence, authentication, fingerprinting, verification, content labeling, protection, and usage control. Most of invisible watermarking methodologies do not work effectively on all types of attacks [3] and various types of applications. The successful watermarking method should show a good performance against intentional [1] and unintentional [2], [3] attacks. Fulfilling invisibility with adaptively desired characteristics and succeeding against various types of attacks are mutually conflicting. Several testing suites have given the evidence, that invisibility and robustness both are very difficult to maintain mutually in available watermarking methodologies.

In the research world, there are two popular techniques for invisible watermarking. First one is spatial domain [4]–[6] techniques. Second one is transform domain [7]–[10] techniques. Both techniques have advantages as well as disadvantages. Therefore, some researchers have used a hybrid method [11] which combines spatial and frequency domains. Classification of invisible watermarking have been made based on several criteria such as robustness, fidelity, embedding type and detection type [3]. Robustness feature refers to the ability to detect the watermark after some digital signal processing operation. Watermarks cannot survive all kinds of attacks. Three major categories can be identified based on robustness: Fragile, Semi- Fragile and Robust. Fidelity feature reflects the invisibility of watermark. It preserves the similarity between the watermarked image and the original image according to human perception. The watermark must remain invisible notwithstanding the occurrence of small degradations in image features, contrast and brightness.

Embedding type describes, the method used for an inserted watermark in the original image. The method used to embed the watermark influences both the robustness against

attacks and the detection algorithm. In literature, three main embedding methods are considered which are the frequency base methods, spatial base methods, and hybrid methods.

Detection type classified determines which resources are necessary for the analysis to extract the watermark from the watermarked image. Based on detection process, can be identified two major categories: Blind and Non-Blind watermark. In the blind watermark detection type the original image and recover data is not available on the receiver's side. Copy control applications use this type of method to send different watermarks to each receiver and the receiver must be able to recognize the watermarked image without the supporting original image or recover data. Non-Blind watermark, in this detection type receiver needs the original data, or any derived information in the detection process. Extraction algorithm needs to refer data about original image and/or recover data.

Steganography [12], [13] techniques have been used to improve the features of watermarking by several researchers. The term steganography is derived from the Greek language and means covert writing. It is a technique of encoding secret information in a communication channel. Steganography is the older method of concealing information. Computer based steganography is one way of data hiding in digital images. It provides data security and integrity as well. The aim is to embed and deliver secret messages in digital images without any suspicion. The secret message can be captioned, plain text, another image, control signal, or anything that can be represented in digital form. The secret message may be compressed and encrypted before the embedding process.

1.3 Problem Statement

A huge number of researches have been done regarding the digital image watermarking. They have used a low number of different theories as base theory and used a huge number of different techniques and processes. After doing a comprehensive literature review, it has been identified, that there are problems and future research directions in watermarking is needed. One major problem of the existing research is to maintain the robustness and invisibility together. Another major issue is self-identification or adaptive extraction of recover data from watermarked image.

Therefore, this thesis has defined the research problem as inadequate attention given to digital watermarking techniques and provided the solution for secure authorship of a digital image by improving the watermark embedding process and adaptive extraction process.

1.4 Hypothesis

It was hypothesized that, the above mentioned problem can be solved by a novel method of invisible watermarking for digital image, based on low level features of the image and Discrete Cosine Transformation (DCT).

1.5 Aim and Objectives

The goal of this study is to develop a robust and adaptive digital image watermarking technique to secure authorship of digital images. The objectives of the study are as follows:

1. Develop a watermark embedding method based on low level features of the images.
2. Develop an adaptive watermark extraction method.
3. Evaluate the robustness, fidelity, and adaptability of novel embedding and extraction methods.

1.6 Low Level Features & DCT Base Approach

This study has divided the entire process into two major parts which are, watermark embedding and watermark extraction or recognition. The main purpose of the embedding process is to hide the recover data into original or host image. The main purpose of the extraction process is to identify the watermark, which has been added in embedding process from watermarked image.

The watermark embedding process is a challenging task, which should ensure fidelity property and robustness property. Edges and corners of the image are important features to represent the content of the image. This study has abstracted a novel model by extending the traditional Harris model and Laplacian of Gaussian (LoG) operator. Novel proposed watermark embedding algorithm shows actual implementation of novel model. The

watermark embedding process has four steps: Identify the features of host image, generate recover data, embed generated recover data, encode.

The first step of the embedding process is to identifying edges and corners of host image. Purpose of identifying edges and corners of the original image is to determine the places to embed the recover data. The second step is generating the recover data according to host image and selected watermark. Watermark object also would be a small image and recover data would be a derivative from pixel intensity of that small watermark image. The third step is to embed generated recover data into edges and corners. Finally, encode or compress watermarked image.

The watermark extracting process is adaptable and decoding algorithm is simple. To extract the recover data the only requirement is the matrix of metadata. Extraction algorithm never requires original image.

The first step of extracting process is to decompress the watermarked image. Thus, it was decided to use traditional JPEG decompress algorithm to decode the watermarked image. Next step is to extract the recover data from watermarked image and reconstruct the watermark object. For this purpose, it was needed to analyses the edges and corners of watermarked image. Then to detect recover data on those edges and corners.

1.7 Structure of Thesis

The rest of the thesis is organized as follows. Chapter 2 critically reviews the literature of digital watermarking and identifies the research problems. Chapter 3 is about the methodology used in this research. Chapter 4 presents experimental design and experimentation. Chapter 5 is an evaluation of the novel solution. Chapter 6 concludes the research with a note on further work.

1.8 Summary

This chapter gave an overall picture of the entire project presented in this thesis. As such, this chapter has described the background/motivation, problem definition, hypothesis, objectives, and a brief overview of the solution. Next chapter presents a critical review of literature on digital watermarking.

Chapter 2

Review of Literature

2.1 Introduction

Chapter 1 has given a comprehensive description of overall project descriptor in this thesis. This chapter provides a critical review of the literature in relation to developments and challenges in digital watermarking. For this purpose, the review of the past researches have been presented under five major sections, namely classification of watermarking, watermarking in spatial domain, watermarking in transform domain, feature base watermarking, steganography methods use in watermarking and evaluation of watermark algorithms. Finally, research problem has defined as the inadequate security in the current watermarking techniques, and identify the feature base watermarking technology as the technique for solving an addressed problem.

Digital images are now widely distributed over the Internet and other media. Nowadays, it is necessary to provide protection mechanisms against unauthorized processing and the use of multimedia contents. There are many techniques, which have been developed to solve this problem. Watermarking is a one important concept can use to avoid unauthorized processing and use of multimedia contents. Other important constituent of information hiding is steganography. Purpose of this section is studying the previous research works of digital image watermarking and steganography.

2.2 Classification of Watermarking

Classification of watermarks can be based on robustness, fidelity, capacity, embedding and detection features. A comprehensive review on digital image watermarking, have been done by Charles W. H. Fung et al. the researchers of Federal University of Technology University, Brazil [1]. The paper has shown, classification of watermarks and proposes of a basic model for watermarking and explained some recent algorithms for image watermarking and their features.

2.2.1 Robustness

Term robustness is highly related with term security. The more roust watermarking method provides more security and assure a higher level of authentication. Robustness feature refers to the ability to detect the watermark after some signal processing operation. Because of this reason robustness become a most important feature of the watermark classification. Another in depth review on watermarking principles and practices have been done by M. Millere et al [2]. This review paper has described the framework of watermarking, properties of watermarking, several attacking types and detection methods. They have more considered noise addition when passing data via communication channel and also, they have discussed the encoded representation of the watermark with the media data. Robust watermarking method provides good resistance against several types of attacks such as low pass filter attack, geometric attack, forgery attack, VQ attack, noise adding, rotation and scaling [4].

All watermark unable to survive in every kind of attacks, hence attacks resilience must be optimized according to application. Different type of applications expects resilience from different type of attacks. Watermark generates and embedding technique defines the surveillance attack types. One watermarking technique may provide protection against one or many attack types. Based on robustness feature, digital watermark able to classify as fragile, semi-fragile and robust [1].

1. Fragile watermark: These types of watermarks can destroy of destruct using simple image processing operation. Some applications want exactly the opposite of robustness [2]. For example, the use of physical watermarks in bank notes. The point of these watermarks is that they do not survive any kind of copying. This property of watermarks calls, fragility. Designing the fragile watermarking methods is easier than designing robust ones. These types of watermark have been used for authentication and integrity verification. Some watermark applications are required to survive certain transformations and be destroyed by others. For example, a watermark placed on a legal text document should survive any copying that does not change the text. But be destroyed if so much as one punctuation mark of the text is moved. This requirement is not met by digital signatures developed in cryptology, which is verify bit-exact integrity, but cannot distinguish between various degrees of acceptable modifications.

2. Semi-Fragile watermark: Some applications are required to survive certain transformations and destructions by others. For example, a watermark placed on a legal text document should survive any copying that doesn't change the text. Purpose of fulfillment of this requirement introduced the semi-fragile watermarks. This type of watermark is able to be surveillance against small intentional modifications. But it will not be surveillance against casual modifications [14]. These types of watermarks have been used in image authentication and tamper control. Semi-Fragile watermarks are more robust than fragile watermarks and less sensitive to classical user modifications. The aim of this method is to discriminate between malicious and non-malicious attack.

3. Robust watermark: According to Patrizio Campisi et al. [15], this type of watermark provides extremely resistant to heterogeneous manipulations. One of the most commonly measuring property is that embedded watermark or watermark signals must be reasonably resilient to various types of attacks and common signal processing operations in digital image processing.

After inserted watermark signal into original image, the distortion may be applied when image distributes across Internet [3]. These distortions can be in many forms. Common distortion types are noise adding, filtering, geometric transformation, rotation and scaling. It is impossible for a watermarking system to be robust against all signal processing operations, whereas the requirement is application subordinate and dependent. For the digital watermarking of images, the good watermarking method is likely to resist against noise addition, filtering processes, geometrical transformations such as scaling, translation and rotation, and compression. These types of marks have been used to copy control monitoring.

As M. L. Miller [2] robustness mentioned two separate issues. First one is whether the watermark is still present in the data after distortion. Second one is whether the watermark detector can detect it. For example, watermarks inserted into images by embedding algorithms, then remain in the watermark signal after geometric distortions such as scaling and rotation. But the corresponding detection algorithms can only detect the watermark if the distortion is first removed. In this case, the distortion cannot be determined and inverted. Detector or extractor cannot detect the watermark even though the watermark is

still present in a distorted form.

2.2.2 Fidelity

Term fidelity is related to visual perception of human. It preserves the similarity between the watermarked image and the original image according to human perception [1]. After adding the watermark signal in the original image, it should not be noticeable to viewer and it should not be sensible to viewer degrade the quality of the content of the original image [2]. Signal may be truly imperceptible. But perceptually based on lossy compression algorithms either introduces modifications. The objective of a lossy compression algorithm is to reduce the representation of data using minimal number of bits. The Lossy compression implies that changing any bit of encoded data should be resulting perceptible difference. Watermark should be detectable after the data compressed and decompressed. The compressed original image data must be different than the compressed watermarked data. This implies that two versions of the data will be perceptibly different once they are decompressed and viewed.

Early works on watermarking [1], [2], [5], [6] focused almost exclusively on designing watermarks that were imperceptible. Therefore, often placed watermark signals in perceptually insignificant regions of the content, such as high frequencies or low-order bits. However, techniques, such as spread spectrum, can be used to add imperceptible or unnoticeable watermarks in perceptually significant regions.

2.2.3 Capacity

The term capacity describes the number of bits of recover data or signal able to insert into original image or object with minimum destruction [1]. In other words capacity is defined using the largest quantity of information that inserted watermarks are capable of hiding [3]. The number of bits that can be inserted through watermarking varies with each application. Embedding algorithm decide the number of bits or size of watermark that can be insert into original object. In case of images, a mark will be a static set of bits. Expect that hidden watermark information not to be destroyed or damaged. In addition, the complexity of the watermark process may be safety related. Because of the attacker will be discouraged to search the insertion in an embedding space and long key position. Therefore, in order to improve the security of the algorithm can enlarge the embedded

space, and increase the size of number of bits into small pieces of the cover image. According to H. Tao et al. [3] and Charles Way et al. [1] , increase the capacity of watermark object, increase the difficulty of attack.

2.2.4 Embedding

Embedding is one main process of every watermarking application. The purpose of this embedding algorithm is insert recover data on the host object. This is most critical part of the entire watermarking application. The method used to embed the watermark influence both the robustness against attacks and the detection algorithm. But some methods are very simple and cannot meet the application requirements. El-Gayyar and von zur Gathen [4] showed that designing a watermark should consider a trade-off among the basic features of robustness, fidelity and payload. There are two main approaches for the embedding process, namely spatial domain and frequency or transform domain.

Spatial watermark insert data in the cover image changing pixels or image characteristics [3], [16]. Spatial domain watermarking algorithm analysis all pixels in host image or object and insert recover data into host image according to embedding algorithm. Spatial watermarking can also be applied using color separation. In this way, the watermark appears in only one of the color bands. This renders the watermark visibly subtle such that it is difficult to detect under regular viewing. Watermark appears immediately when the colors are separated for printing. This renders the document useless for the printer unless the watermark can be removed from the color band. Spatial domain watermarking will be described in section 2.3.

Transform or frequency domain algorithms [17]–[19] hide the watermarking data in transforming coefficients, therefore spreading the data through the frequency spectrum, making it hard to detect and strong against many types of signal processing. Transform domain watermarking will describe in section 2.4.

2.2.5 Detection types

Detection type determines which resources are necessary for the analysis and extract the watermark from the watermarked image. Another word for detection is extraction. The watermark extraction is an important process of every watermarking application. The

watermark extraction process should be simple and adaptive [1], [2]. There are two main approaches for the extraction process, namely blind and non-blind.

1. Blind detection: In this detection type, the original image and mark data is not available to the extraction algorithm. For example: Copy control applications must send different watermarks for each user and the receiver must be able to recognize and interpret these different marks.

2. Non-Blind: In this detection type, the extraction algorithm needs the original data, or any derived information from it, for the detection process. This data will also be used in the extraction algorithm.

2.3 Watermarking in Spatial Domain

The spatial domain watermarking technique is a popular method in watermarking research world. Many researchers [4]–[6], [20]–[24] have done their research works based on spatial domain. In spatial domain, the watermark bits directly added to the pixels of the cover image. Spatial domain methods can be easily modeled and analyzed mathematically. However, the embedded watermark can be easily destroyed or removed by signal processing attacks such as filtering [20]. The spatial domain technique makes use of the human visual system. But sensitive to the image scaling [4], [5] therefore that same information must be embedded again and again in different locations of the host image. The least significant bit (LSB) [23] method is an example of a spatial domain method where the watermark is embedded into the least significant bits of the cover image. In this method, first the stream of bits is extracted from the watermark and then shifted to the right. The shifted bit planes are added to the least significant bits of the cover image to get the watermarked image. The least significant bits are highly sensitive to noise, therefore that the watermark can easily be removed by image manipulations such as rotation and cropping. Thus, the LSB method provides high imperceptibility and less robustness.

The correlation based method is another example of spatial domain techniques [24]. In this method, the watermark is converted into pseudo noise sequence, which is then weighted and added to the cover image bits. The watermarked image is compared with the cover image to detect the inserted watermark. The spatial domain methods are less complex

compared to transform domain methods, however weak to different image attacks. The data hiding capacity of spatial domain techniques is higher than that of transform domain methods. Spatial domain techniques offer higher robustness to geometrical transformations.

Nikolaidis and Pitas [20] have done the study of robust image watermarking in the spatial domain. They proposed the method for watermarking in the spatial domain by slightly modifying the intensity of randomly selected image pixels. The detection process does not require the existence of the original image. Detection algorithm comparing the mean intensity value of the watermarked pixels against those pixels not watermarked. They have introduced a concept called image dependent watermarks through this study. After doing by evaluation they have concluded proposed method can successfully implement on JPEG images [25]–[27]. In 1992, JPEG became an international standard for compressing digital still images. The acronym JPEG comes from the Joint Photographic Experts Group. Members of the International Organization formed JPEG in the 1980's for Standardization (ISO) and the International Telecommunications Union (ITU).

From 1990, a standardization effort known by the acronym JPEG, for Joint Photographic Experts Group, has been working toward establishing the first international digital image compression standard. As Gregory K. Wallace [26] JPEG's goal has been to develop a method for continuous-tone image compression, which meets the following requirements.

1. Should be near the state of the art with regard to compression rate and accompanying image fidelity, over a wide range of image quality ratings. Specially in the range where visual fidelity to the original is characterized as "excellent". The encoder should be parameterizable, therefore the application can set the desired compression/quality trade off.

2. Should be practically applied to any kind of continuous-tone digital source image and not be limited to classes of imagery with restrictions on scene content, such as complexity, range of colors, or statistical properties.

3. Should have tractable computational complexity, to make feasible software

implementations with viable performance on a range of CPU's, as well as hardware implementations with viable cost for applications requiring high performance.

4. Should have the following modes of operation: sequential encoding, progressive encoding, lossless encoding and hierarchical encoding.

JPEG has used mathematical theory called Discrete Cosine Transformation (DCT). 2D DCT simply denoted by $F^{-1}(u,v)$, which used for JPEG compression. JPEG have four main steps, namely preprocessing, transformation, quantization, and encoding. DCT and JPEG will describe in detail methodology chapter.

The JPEG2000 [27] standard provides a set of features that vital importance to many high-end and emerging applications, by taking advantage of new technologies. It addresses areas where current standards fail to produce the best quality or performance and provides capabilities to markets that currently do not use compression. JPEG2000 has used mathematical theory called Discrete Wavelet Transformation (DWT).

Spatial domain watermarking algorithms have shown, robust against compression, filtering, cropping, and scaling. Francesc et al. [21] have done the study and shown above characteristics in spatial domain watermarking. The researchers presented two image watermarking algorithms for copyright protections. The first one is robust against compression, filtering and cropping and the second one for visual components and robust against compression, filtering, scaling and moderate rotations. Some researchers proposed first embed the watermark directly into the spatial domain while the other embeds the watermark after converting to the DCT Domain. Heather Wood [28], a researcher of Adams State College, Colorado has been done study of invisible watermarking for color images following above approach. This study based on spatial domain techniques and discrete cosine transformation (DCT). The extraction methods are able to detect the watermark, perfectly with no attacks. They have suggested future research in resistance to geometric attacks such as cropping and scaling.

Literature provide evidence, different researchers have used different places of image to embed the recover data. Princeton University and Microsoft Research [22] have done a

research based on triangle meshes representing surfaces in 3D. In proposed watermarking process first generate the watermark vector, then surface basis functions used to embed the watermark vector into the mesh. In the extraction process, first need to convert all frames into same coordinate, and then check for watermark presence in a suspect mesh. They have proven to be robust against a wide variety of attacks, including vertex reordering, addition of noise, similarity transforms, cropping, smoothing and insertion of a second watermark over watermarked image. Researchers have suggested the future research on fast computation, use most other surfaces, and improve algorithm against the other type of attacks such as general affine transforms, projective transforms, free-form deformations, etc.

Literature review has provided evidence that spatial domain methods are less complex, but weak against different types of image processing attacks. The data hiding capacity of spatial domain techniques is higher than that of transform domain methods. Spatial domain techniques offer higher robustness to geometrical transformations.

2.4 Watermarking in Transform Domain

Transform domain watermarking technique is another popular method in watermarking research world. Many researchers [7]–[11] have done their research works based on transform domain. The robustness and imperceptibility of the watermarked images can be improved by performing watermarking in the transform domain. Transform domain techniques can provide better robustness against compression and filtering attacks, because of the watermark coefficients spread throughout the cover image. In transform domain, modifying the image coefficients using image transforms does watermark embedding. Masking techniques based on transform domain are more robust than spatial domain methods with respect to cropping, compression, noise adding and filtering attacks. The main advantage of masking techniques is that they embed watermark coefficients in large areas of the host image.

A research on wavelet based digital image watermarking has been done by Wang et al. [7]. This study proposed novel an adaptive watermark embedding and retrieval method. This research based on Multi Threshold Wavelet Codec (MTWC). A proposed method of this research is first determining significant wavelet sub-bands and then selects a couple of

significant wavelet coefficients in these sub-bands to embed watermarks. The watermark retrieval techniques can detect the embedded watermark without the help from the original image. They have experimentally proved embedded watermark is robust against various signal processing and compression attacks.

Discrete Cosine Transformation (DCT) is a most prominent mathematical technique used in transform domain watermarking. DCT is one of the most common linear transformation techniques in digital signal processing. Discrete Wavelet Transformation (DWT) is widely used mathematical theory in transform domain watermarking. Combination of DCT and DWT gave the better surveillance against several types of attacks. Mei Jiansheng and colleagues [8] have given the evidence to the above statement and presented digital watermarking algorithm based on discrete cosine transformation and discrete wavelet transformation. Here used 2D-DCT for collecting and convert the main information of the original image into the smallest low-frequency coefficient. The idea behind DWT in image processing is multiple differentiated decompose the image into sub image. This research proposed the method, find the wavelet coefficient in the high frequency band of original images using DWT and find the low frequency coefficient of watermark using DCT. Then generate a watermark combing these high-frequency component and low-frequency component. After using Inverse DWT (I-DWT) generate a watermarked image. Apply DWT technique on the original image and watermark image both and compare the distill signal, and then use inverse DCT (I-DCT) for detection watermark. They have theoretically and experimentally proved the robustness against many common image processing operations of filers, sharp enhancing, adding salt noise, image compression, image cutting and so on.

Many researchers [29]–[31] have used wavelet-base watermarking algorithm for their studies. Wavelet is an oscillatory function of finite duration. The wavelet provides both spatial, and frequency description details of the image. The temporal information is retained in this wavelet transformation process compared to other transforms like DCT and DFT. Haar, Daubechies, Complex, Balanced, Stationary, Morphological, Non - tensor, Berkley, Mexican-hat, Morlet, Shannon and Bi-orthogonal are the different wavelets used to perform image processing. The DWT is not effective to analyze non-stationary signals [29]. Whereas short time Fourier Transform is an effective tool to do that operation, but

the drawback is that it gives a constant resolution at all frequencies. DWT provides both spatial, and frequency description of an image with multi-resolution. The multi-resolution property of the wavelet transform can be used to exploit the fact that the response of the human eye is different to high and low frequency components of the image.

A research on efficient wavelet-base watermarking algorithm has been conducted by Xiaojun Qi a researcher of Utah State University [29]. Proposed watermarking algorithm of this research is embeds a binary logo watermark by modifying the appropriate sub band images in the wavelet domain. The watermark detection algorithm has the nice advantage of proposed method, without referring to the original image it can detect the watermark. This paper evaluates own approach and confirm effectively hide a robust watermark due to the exploitation of the characteristics of the human visual system. This proposed method guaranteed robust to a variety of image processing techniques, such as JPEG compression, sharpening, resizing, and geometric operations.

DWT can be applied to an entire image without using block structure as used by the DCT, thus reducing the blocking artifact. Wavelet is an oscillatory function of time or space that is periodic and of finite duration with zero average value. Wavelets can be generated by dilating and translating mother wavelet. Wavelet provides time - frequency representation of a signal. It used to analyze non-stationary signals. A research on digital watermarking has been done by Shanjun Zhang and Kazuyoshi Yoshino the researchers of Kanagawa University, Japan [30]. This study introduces a novel watermarking method to embed QR codes in digital images. This research based on mathematical theory of discrete wavelet transformation. These authors have implemented the solution by original image is divided into blocks, and QR codes are added to particular bits of LL2 level coefficients of the selected block. The proposed embedding method of this study, original image is divided into a set of 4x4 blocks. Each block is transformed into level two with Daubechie mother wavelet function. Watermark signals are the embedded in the low frequency domain of the blocks. They proposed four step process to extract the watermark. The first step is watermarked image are divided into 4x4 blocks. The second step is used discrete wavelet transform, and then the low frequency component of the block is compared with neighboring blocks. Third is calculating the sum of horizontal and vertical values is greater than a predefined threshold then it extracts as watermark bits. Fourth step is

constructing the QR code using above bit series. By the way can't see an evaluation of the proposed method in this paper.

Multi-resolution technique has used in wavelet transform where different frequencies are analyzed with different resolutions. Big wavelets give an approximate value of a signal, while the smaller wavelets boost up the smaller details. DWT is computed either by using convolution based or lifting based procedures. In both the methods, the output sequence decomposed into low pass and high pass sub bands, where each sub band constituting of half the number of samples of the original sequence. The DWT represents an NxN image by N2 coefficients. The DWT can be implemented through a filter bank or lifting scheme. The DWT of an image is analyzed by allowing it to pass through an analysis filter bank followed by down sampling. The analysis filter bank consists of low pass and high pass filters at decomposition stage. When an image passes through these filter banks, the image split into two sub bands. The low pass filter performs averaging operation and extracts the coarse information of the image. Whereas the high pass filter performs difference operation and extracts the details of the image. Then two down samples the output of the filtering operation. This operation splits the image into four sub bands, namely, LL, LH, HL, and HH.

Nafornita and Alexandru Isar [31] have used statistical characteristics of DWT. The proposed method of this research, decompose into sub bands using DWT and select the coefficient to embed as a watermark by doing the statistical analysis of those coefficients. They have tested method against different types of attacks such as lossy compression, scaling, cropping, intensity adjustment, filtering and collusion attack. Pixel-wise masking based method on local standard deviation and wavelet compression has created perceptual watermark by using a local standard deviation of the original image and then compressed in wavelet domain. They have evaluated this method and assured the imperceptibility of watermark. After continue the study, they have proposed improved pixel wise masking in a better way. This novel proposed method use high frequency component of sub image for appreciating the luminance content. They have tested and confirm this method work better on human visual system behaviors. Same researchers again have proposed a novel method of increasing robustness using turbo codes [32]. In this paper, they have presented a watermarking system that uses the bi-orthogonal discrete wavelet transform and the

message is encoded before embedding. Based mathematical theory of this research is DWT and duo-binary codes. They have proposed four steps novel approach in this paper. The first one is turbo coding of the watermark message. The second step is embedding the turbo coded watermark into the host image using a perceptual mask. The extraction of the turbo coded watermark from the watermarked image is third step. The last step is identifying the possibly corrupted image.

Singular Value Decomposition (SVD) is the powerful numerical analysis tool used to analyze matrices, where the image matrix can be decomposed into three matrices that are about the same size as the original image matrix. SVD transformations preserve both one-way and non-symmetric properties, usually not available at DCT and DFT. The use of SVD in digital image watermarking has advantages like the size of the matrices not fixed and can be either rectangular or square. The performance of wavelet based watermarking algorithms has been improved by using different optimization techniques such as singular value decomposition, independent component analysis, the support vector machine, genetic algorithm, artificial neural network and fuzzy logic,etc. Some researchers have conducted their research on Singular Value Decomposition (SVD) and Lifting Wavelet Transform (LWT) [11]. The hybrid watermarking algorithm based on SVD and LWT, does not require to use human visual system characteristics. In this research, they decomposed the host image into a two-level using LWT and compute the sub band using using inverse LWT. They have experimentally proved this method is robust against several attacks such as: noise addition, histogram equalization, gamma correction, JPEG compression, cropping, rotation and random line and column removal. After done continue study of this method they proposed improved security ownership mechanism of digital watermarking based on SVD. They have continued the research and then proposed a novel technique, Multi-Objective Genetic Algorithm Optimization [10] for Image Watermarking based on SVD and LWD. This research work is improvement of previous research by implementing using genetic algorithm. Researchers proposed another novel image watermarking algorithm based on Multi-Objective Ant Colony Optimization (MOACO) and SVD in wavelet domain. Novel proposed method is binary watermark decomposed using SVD. Then the singular values are embedded in a detailed sub-band of host image. Experimental results of evaluation section have shown improved performances, transparency and robustness of the method.

The research based on the multiple transform method, which is the discrete wavelet transforming and Discrete Fractional Random Transform (DFRNT) [17] has done by Kim et al. They have used two-dimensional bar-code for hiding information and apply the block code encoding to generate a watermark. Then watermark added to host image using DWT-DFRNT quantization technique. They prove watermark generated by this novel approach is more difficult to break and more robust. This research provides advance and robust algorithm against general image processing attacks such as image compression and noise adding. Dual transform domain watermarking algorithm based on DCT and DWT has used in [18] here. This paper analyzes the embedding position and strategy of transform domain algorithms. The DC coefficient of the original image is divided into blocks of DCT spectrum, and then combined with DWT coefficient in this method. This particular research has the main advantage of extraction algorithm and it has self-recovery ability.

Zhao Dawei and coworkers have presented digital watermarking technique based on chaos wavelet domain [19]. They transform sub images locally and embed, which is extracted from the original image using DWT. Watermark detection algorithm computing the correlation between the watermarked coefficients and the watermarking signal. They have simulated the results proved high fidelity and high robustness of proposed method, especially under the geometric attack. Ayubi et al. has been conduct research on Chaos Based Blind Digital Image Watermarking in The Wavelet Transform Domain [33]. They have increased the security of watermarked image by doing increasing the length of key of chaotic maps. Additionally, they have upgraded mapping method determines the location of DWT coefficient were embedded. They developed the simulation process and it results indicate that the new algorithm can preserve the hidden information against geometric and non-geometric attacks.

The success of wavelets is mainly due to the good performance of piecewise smooth functions in one dimension. Unfortunately, such is not the case in two dimensions. In essence, wavelets are good at catching zero-dimensional or point singularities. But two-dimensional piecewise smooth signals resembling images have one-dimensional singularities. That is smooth regions are separated by edges, and while the edges are discontinuous across, they are typically smooth curves. Intuitively, wavelets in two

dimensions are obtained by a tensor-pro duct of one dimensional wavelets and they are thus good at isolating the discontinuity across an edge, but will not see the smoothness along the edge. To overcome the weakness of wavelets in higher dimensions, Minh and Marting [34] have done research of representations image in Ridgelet Transform (RT), which deal effectively with line singularities in 2D. RT technique has introduced to the digital watermark by Campisi, Kundur and Neri [15]. The RT allows most significant coefficients represent the most energetic direction of an image with straight edges. This transformation allows representing edges and other singularities along curves in a more efficient way. They have experimentally proved; the proposed method is high robustness to most attack types and maintaining an excellent perceptual invisibility.

2.5 Feature Base Watermarking

A digital image is a matrix of many small elements, or pixels. Each pixel is represented by a numerical value. In general, the pixel value is related to the brightness or color that will see when the digital image is converted into an analog image for display and viewing. In machine learning, pattern recognition and in image processing, feature extraction starts from an initial set of measured data and builds derived values (features) intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps, and in some cases leading to better human interpretations. Feature extraction is related to dimensionality reduction.

During literature study, founded the low-level features of digital image have used for invisible watermarking. Edges, corners, blob, and ridge are low level features, which are used into digital watermarking. Chris Harris [35], a famous researcher in the image processing field has done the comprehensive research inverted very popular Harris corner detection method to the world. He has improved the Moravec's corner detector by considering the differential of the corner score with respect to direction directly, instead of using shifted patches. This corner score is often referred to as autocorrelation, since the term is used in the paper in which this detector is described. However, the mathematics in the paper clearly indicates that the sum of squared differences is used. This research has purposed step by step process for find the corners. First step is finding the edges of horizontal and vertical directions. Then find the cross product of those edges. Then use the Gaussian operator to remove the noise. Finally, non-maximal suppression has done. Harris

operator is a fundamental technique of digital image processing.

David G. Lowe, the famous researcher of the University of British Columbia has conducted key research of distinctive image features from scale-invariant key-points [36]. This research presents a method for extracting distinctive invariant features from images, which can be used to perform reliable matching between different views of an object or scene. The features are invariant to image scale and rotation, and are shown to provide robust matching across at a substantial range of affine distortion, change in 3D viewpoint, addition of noise, and change in illumination. This research is very important & key research of the image processing field. This research describes an approach to using these features for object recognition. The recognition proceeds by matching individual features to a database of features from known objects using a fast-nearest neighbor algorithm, followed by a Hough transform to identify clusters belonging to a single object, and finally performing verification through least-squares solution for consistent pose parameters. This approach to recognition can robustly identify objects among clutter and occlusion while achieving near real-time performance. Feature detection and matching are crucial for robust and reliable image registration. Although many methods have been developed, they commonly focus on only one class of image features. The methods that combine two or more classes of features are still novel and significant. Mexican hat [37] function-based operator has used for image feature detection, including the local area detection and the feature point detection.

The edge histogram descriptor (EHD) is one of the widely-used methods for shape detection. It basically represents the relative frequency of occurrence of 5 types of edges in image block. This edge histogram descriptor has used to digital watermarking [38]. EHD can be efficiently utilized for image matching. In [38] this research, they have to increase the matching performance, by using global, semi-local and local edge histograms. They have used the spatial domain technique and sub divided original image into sub images. Then identify the edge location and edge type, which are more suitable to hide recover data. Won et al. [39] have proposed a novel watermarking method based on EHD, purpose of increased the robustness against re-sizing, compression, sharpening, image cutting and so many attacks. Edges in images constitute an important feature to represent their content. Also, human eyes are sensitive to edge features for image perception. One way of

representing such an important edge feature is to use a histogram. An edge histogram in the image space represents the frequency and the directionality of the brightness changes in the image. It is a unique feature for images, which cannot be duplicated by a color histogram or the homogeneous texture features. One limitation can identify of this work, proposed method can be used only in MPEG-7 image description format.

2.6 Steganography Methods Used in Watermarking

Steganography is a Greek term meaning of "covered writing". Use of Steganography is hiding a secret message within an ordinary message and the extraction of it, at its destination. In modern digital steganography, data is first encrypted by the usual means and then inserted, using a special algorithm, into redundant data that is part of a file format such as a JPEG image. Think of all the bits that represent the same color pixels repeated in a row. By applying the encrypted data to this redundant data in some random or non-conspicuous way, the result will be data that appears to have the noise patterns of regular, non-encrypted data. In literature gave the evidence [12], [13], [40]–[43], this steganography methods have used in digital image watermarking.

The researcher of National Kaohsiung First University, Thaiwan [12] has done a study about steganography method for hiding the data in gray-valued images called Pixel Value Difference (PVD). This research based on texture block coding method. The authors have implemented the solution by using pixel-value difference of non-overlapping blocks of the cover image. This method provides more imperceptible and less sensitive stego-image than LSB replacement methods. Another advantage of this method is secret image can be extracted from stego-image without referencing original the cover image. They have tested this method done by several experiments and results show the feasibility of novel methods. They are calculating the difference between gray-value of the two pixels in each block and categorize into number of ranges. This range interval selected based on human vision's sensitivity. This method utilizes the characteristic of the human vision's sensitivity. Chang and colleagues [13] have extended above method by using Tri-way Pixel Value Differencing (TPVD). This research based on pixel-value difference (PVD) on two consecutive pixels []. They extend the PVD method through three different directions. This particular research has advantages such as hide more secret data into cover image than PVD method and it can extract from stego-image without referencing original

image. As well as this method reduce the quality distortion of the stego-image.

Watermark embedding algorithm using steganography has introduced by Jessika Fridrich [40] for raster digital images. This research based on stochastic modulation. They embedded secret message as week noisy signal using probabilistic manner. This research has advantage of more secure schemes because of steganalysis [41] have many difficulties to identify noisy signal. Anyway, extraction algorithm has difficult to identify secret message and device noise separately. A research on reducing the computational cost of additive noise steganalysis has been conduct by a research groups in Rensselaer Polytechnic Institute, Troy, NY and Carnegie Mellon University, Pittsburgh, PA [42]. This study based on three-dimensional Fast Fourier Transform (FFT) [43]. This transforms method is very fast and powerful. This research has the advantage of reducing the required processing power of noise steganalysis. Anyway, they ware assumes the stego-image is created by adding a pseudo-noise to a cover image. But no way to get confirmation stego-image created by adding a pseudo-noise or another method.

A comprehensive research of DCT coefficient dependent quantization table modification steganographic algorithm, have done by Lochandaka et al. [44] the researchers of the university of Moratuwa. This study presented a new data hiding technique based on the DCT coefficients and modified quantization table values. Embedding strength of each coefficient are determined by the mathematical formula which compared to the DCT coefficient and appropriate quantization table value in order, then the secrete bits are embedded into frequency components of the quantized DCT coefficients using least significant bit (LSB) method to enable large embedding capacity without image degradation. This research has used 8x8 JPEG block and have future development to 16x16 and 32x32 block sizes.

On same token, many, research has been done for digital image watermarking and digital image steganography. Until now, this review concerned to the spatial domain techniques, transform domain techniques, feature base watermarking and steganography used in watermarking.

2.7 Evaluation of watermark algorithms

Digital watermarking is an efficient solution for copyright protection of digital images, which inserts copyright information into contents itself. This information is used as evidence of ownership. Digital watermarking has many applications, in which robustness has been an important issue. There have been many watermarking researches inspired by methods of image coding and compression. Most algorithms perform well against signal processing attacks. Nevertheless, in blind watermarking, these algorithms show severe weakness to geometric distortion attacks that desynchronize the location of the inserted watermark information and prevent watermark detection.

Degradation of the original image information is natural after applied any watermark embedding algorithm. How compare the information degradation between original image and watermarked image? It is an important question. Various statistical methods have used in literature for evaluating this degradation. Mean Square Error (MSE), Peak Signal to Noise Ratio (PSNR), Normalized Cross-Correlation (NCC), Average Difference, Maximum Difference, and Normalized Absolute Error (NAE) [45] are mostly used mathematical concept to measure the degradation after adding watermark into original the image.

There have been many feature extraction techniques in image processing, which can be used in watermarking applications. In [46] has described information degradation, when used various feature extraction techniques. Here described theoretical and experimental evaluation result of Harris Corner Detection, Mexican Hat Wavelet Scale Interaction, and Scale-Invariant Key-point Extractor. An objective image quality metric can play a variety of roles in image processing applications. First, it can be used to dynamically monitor and adjust image quality. Second, it can be used to optimize algorithms and parameter settings of image processing system. Objective image quality metrics can be classified according to the availability of an original image, with which the distorted image is to be compared. Most existing approaches are known as full-reference, meaning that a complete reference image is assumed to be a known one. In many practical applications, although the reference image is not available, a no-reference or “blind” quality assessment approach is desirable. In a third type of method, the reference image is only partially available, in the form of a set of extracting features made available as side information to evaluate the

quality of the distorted image. This is referred as reduced-reference quality assessment. This paper focuses on full-reference image quality assessment. The simplest and most widely used reference quality metric is the mean squared error (MSE), computed by averaging the squared intensity differences of distorted and reference image pixels, along with the related quantity of peak signal-to-noise ratio (PSNR). These are appealing because they are simple to calculate, have clear physical meanings, and mathematically convenient in the context of optimization. But they are not perfectly matched to perceive visual quality. Natural image signals are highly structured. Their pixels exhibit strong dependencies, especially when they are spatially proximate, and these dependencies carry important information about the structure of the objects in the visual scene. SSIM is a most suitable evaluation method for natural image [47].

Multi-scale Structural Similarity Index Matrix (MS-SSIM) has used to calculate the degradation of watermarked image in [48], [49]. They have proposed a multi-scale SSIM method for image quality assessment. MS-SSIM takes the reference or original image and distorted or watermarked image signals as the input, the system iteratively applies a low-pass filter and down samples the filtered image. They have tested the number of image quality assessment algorithms and experimentally proven that the advantages of MS-SSIM. A novel concept has introduced Lin Zhang et al. [50] for calculating the degradation of image. This method, called Feature Similarity Index Matrix (FSIM) for image quality assessment. This novel feature-similarity index matrix proposed based on the fact that the human visual system (HVS) understands an image mainly according to its low-level features. Specifically, a dimensionless measure of the significance of a local structure is used as the primary feature in FSIM. FSIM is play complementary roles in characterizing the image contents. Experimental results on benchmark databases show that FSIM can achieve much higher consistency with the subjective evaluations than all the state-of-the-art IQA metrics used in comparison.

2.8 Problem Definition

So far discussion about digital watermarking techniques shows that spatial domain techniques and frequency domain techniques, both can use to generate the watermark on the digital image. Some researchers [4], [20], [21] have used spatial techniques for this purpose. But many researchers [8], [17], [18], [28], [30], [31], [34] have used

transformation techniques as well. As the literature review since, transformation techniques are most applicable. But spatial techniques also useful and it has several advantages in the watermark detection process. I have been done depth study of previous research work. The most of the researchers [8], [28], [51] have been used Discrete Cosine Transformation (DCT) and Discrete Wavelet Transformation (DWT) [17], [30], [52] as base mathematical theory.

The same mathematical theory had used by several different studies. So many researchers have been used wavelet transformation theories in the different manner. Some of the researchers [5] used significant coefficient of wavelet sub-bands as watermark. This sub-band selection method and selected sub-band also varied research by research. Some researchers have used the wavelet coefficient of the same image as watermark. Some of researcher [45] has used another image or QR code or another bit pattern as watermark. Some of researcher have used difference image as watermark. Another researcher has used chaos wavelet coefficient as watermark. Combination of singular value decomposition (SVD) and wavelet transformation have used for generate watermark. Another research has been based on Discrete Fractional Random Transform (DFRNT) and Discrete Wavelet transformation (DWT). Each of above techniques provides different level of robustness, fidelity, capacity and security features of watermarking.

The watermark detection process is very important part of digital watermarking. Detection methods are highly dependent on how embedded the watermark. Large number of proposed methods required original image in the detection process. But the less number of proposed methods are not required original image in the detection process. When considering spatial domain techniques, these have been used comparatively low number of researchers. These methods are robust against compression, filtering, scaling and moderate rotations. Some of researcher [11], [28] have used a combined method of transform domain and spatial domain.

Steganography or covered writing techniques can be used for digital image watermarking. So far discussion about the image steganography techniques shows that transformation methods and probabilistic methods have been the main technologies for steganography and digital watermarking solutions. Many researchers have used transformation as the

basic method for this purpose. But some researchers have used probabilistic methods as well. However, difficult to find integrated solutions with transformation and probabilistic both. Since, transformation methods are most applicable. But probabilistic methods also useful and it has several advantages in security wise.

The literature review has identified various unsolved problems including security, efficiency and reliability of watermark algorithms. Table I summaries the achievements and the limitations of the key research discussed in this chapter.

TABLE I. SUMMARY OF LITERATURE OF WATERMARKING

Researchers	Technology used	Key benefits	Limitations	Remarks
Wang et al.	Threshold wavelet codec	Robust against compression attack	Limited extendibility	Better to improve the extendibility and usability
Mei Jiansheng et al.	DCT & DWT	Robust against filtering attack	Limited extendibility	Better if can improve the extendibility and usability
DK Park et al.	EHD	Robust against re-sizing, compression, and image cutting	Apply only on MPEG-7	Good data hiding provided but it's better to improve the usability
Xiaojun Qi	DWT	Robust against compression & filtering attacks	Difficult to extract	Should improve the extraction method
Shanjun Zhang and Kazuyoshi Yoshino	DWT and QR code	Provide high robustness capabilities	Can't see the evaluation	Proper evaluation is needed. Better if can fine tune extraction process
Corina Nafornita and Alexandru Isar	DWT	Robust against compression & crop	No proper extraction method	Need to consider about the extraction process
K. Loukhaoukha et al.	SVD & LWT	Good transparency	Difficult to implement	Better to improve the extendibility and usability

M. Kim, D. Li, and S. Hong	DWT-DFRNT	Resistance to compression and noise adding	Poor detection method	Good data hiding provided but it's better to improve the watermark detection process
Heather Wood	DCT	Low resistance	Good detection method	Need to improve robustness
Chang et al.	TPVD	Extract without referring original image	Reduce quality of image	Really good extraction provided.

So far discussion, can see the huge number of researches have been done regarding digital image watermarking and steganography, they have been used less number of different theories as based theory and used the huge numbers of different techniques and approaches. After doing a comprehensive literature review, there have identified still have some problems and future research directions in watermarking. One major problem of existing research is security of embedded watermark. Another major issue is self-identification or adaptive extraction watermark from watermarked image.

Therefore, this thesis has to define the research problem as inadequate attention given to the digital watermarking techniques and find the solution for secure authorship of the digital images by improving the watermark embedding process and adaptive extraction process.

2.9 Summary

This chapter presented a comprehensive literature review of the digital watermarking research and identified the research problem as the inadequate attention to adaptability of digital watermarking algorithms. Here has been identified the feature based watermarking techniques can improve the adaptability of watermarking and solve the above problem. Next chapter will discuss the methodology, which has used in novel solution.

Chapter 3

Methodology of Watermarking

3.1 Introduction

Chapter 2 has given a comprehensive literature review of digital watermarking. This chapter describes a novel approach to invisible digital watermarking. During above literature review, has found a number of different technologies used in digital watermarking. It was conducted the research in two main paths. The first one is pixel based methods and the second one is feature based methods. This chapter describes these two techniques in detail.

This research was conducted by using different theories and a number of different watermarking methods. It considered two main techniques: Pixel based techniques and feature based techniques. Discrete Cosine Transformation (DCT) has been used with both the above methods for the purpose of compression. All the methods have used two objects. The first one is the host or original image and second one is the watermark image. Host image can be raw-image, bitmap image or JPEG image. Watermark image should be small compressed or uncompressed image.

The pixel based methods, analyses the intensity values of the original image and watermark image. Then generate the recover data according to intensity values of the original image and watermark image. Then insert recover data into original image and compress using DCT. Feature based methods, analysis the low-level features of the original image. Then select the most immutable feature in the original image and generate the recover data according to intensity values of which features points of the original image. Then insert recover data into original image and compress using DCT.

The watermark extraction process is simple and requires minimum information. Pixel based techniques have a common algorithm for extraction process independently on embedding method. Feature based techniques have loosely depended extraction algorithm on embedding method.

3.2 Pixel Based Techniques

This section presents the novel pixel based solutions to address the research problem. This method, analyses the intensity values of the original image and watermark image. Then generate the recover data according to intensity values of original image and watermark image. Generated recover data is inserted into original image using an embedding algorithm and finally it is encoded using compression algorithm based on discrete cosign transformation. Watermark extracting process is adaptable and decoding algorithm is simple. The first step of extracting process is decompressing the watermarked image. Thus, it was decided to use traditional JPEG decompress algorithm based on DCT, to decode the watermarked image. Then it has to be extracted the recover data from watermarked image and reconstruct original or host image. Inputs of this extraction and recognition process are watermarked image and metadata file generated by embedding process. Appendix A, Figure 1 illustrates the high-level design diagram of pixel base watermarking techniques.

3.2.1 Pixel Based Embedding

Watermark embedded process, goes through each and every pixel of the host image and embed watermark image data into those pixels without considering any feature of host image such as edges or corners. This embedding process, adds the magnitude value of difference between host image and watermark image into the pixel of host image. This study has developed three techniques and experimentally test based on direct pixel manipulation. The first one is random insertion. The second one is to insert into less sensitive points to human vision. The third one is to insert into a least significant point in JPEG macro-block. Figure 3.1 illustrates the high-level design diagram of pixel base watermark embedding technique.

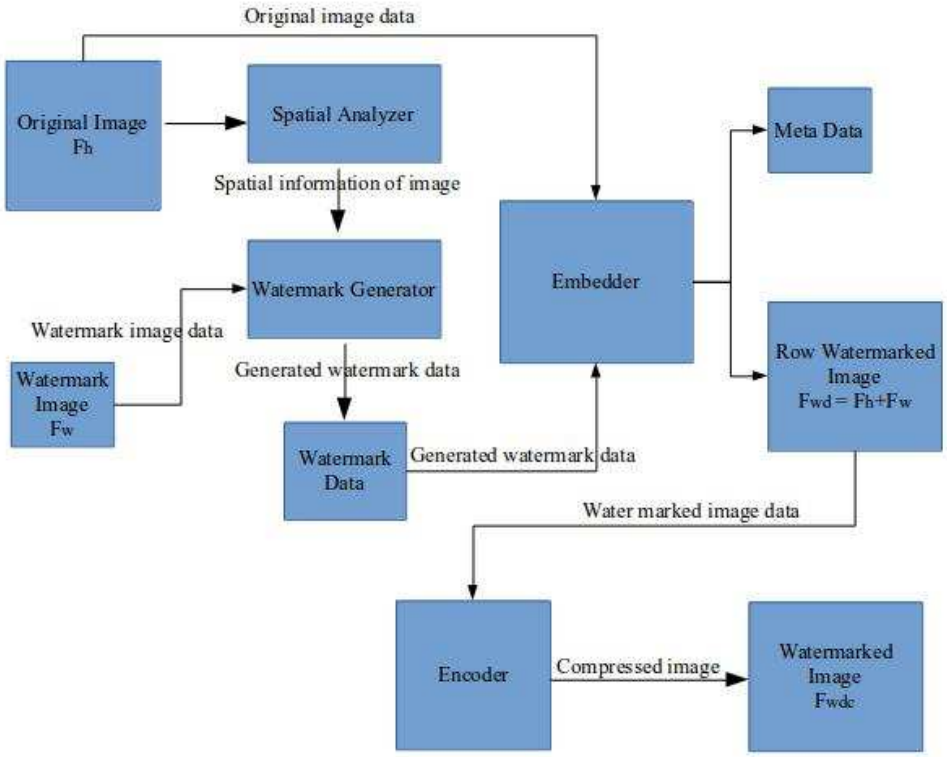


Figure 3.1: High level design diagram of pixel base watermark embedding technique

Inputs of the pixel based watermarking system are original or host image and watermark. Original image is a compressed or uncompressed raw-image. Normally original image is a large image. The watermark is a small compressed or uncompressed image. This experiment has used 48x48 color or grayscale image as a watermark.

Original image is sent to spatial analyzer and spatial analyzer analyses the intensity values of each pixel in the original image. In this case, a separate color component of each pixel has been considered. Thus, here red, green, and blue components separately have been taken. I(x,y) denoted intensity value of pixel (x,y). Ir(x,y), Ig(x,y), Ib(x,y) are color component, which red, green, and blue color components respectively.

$$\overbrace{Ir(x,y), Ig(x,y), Ib(x,y)}^{I(x,y)}$$

(3.1)

The spatial analyzer sends signals to the watermark generator. Another input of watermark generator is a watermark image. The watermark generator generates the recover data according to watermark image and signal which is sent by spatial analyzer. Recover data, calculates using spatial information of the original image and spatial information of the watermark image. Then recover data computes using the difference between original image intensity values and watermark image intensity values. Finally, it has taken the square root of absolute magnitude value between original image and watermark image. Equation 3.2 represents, how to calculate recover data for one pixel.

$$W(x,y) = \sqrt{|O(x,y) - w(x,y)|}$$

(3.2)

Where W(x,y) is recover data for one pixel. O(x,y) and w(x,y) are intensity value of individual pixel of the original image and watermark image. Above calculation has been applied to ever color components red (Ir(x,y)), (Ig(x,y)), and (Ib(x,y)). Then the complete recover data has been compute as equation 3.3.

$$W = \sum_{i=0}^{i=nw} w\,(i)$$

(3.3)

Complete recover data denoted by ***W***, and recover data correspond to each pixel denoted by ***w(i)***. Number of pixels in watermark image denoted by ***nw***. After applying the above formula into every color band, the resulting recover data matrix represents the following.

TABLE II. RECOVER DATA MATRIX OF PIXEL BASED METHODS

Pixel	Y	X	Wr(x,y)	Wg(x,y)	Wb(x,y)
1	22	5	2	3	6
2	87	56	7	3	4
3	112	235	3	5	1
4	198	453	9	1	4
5	322	502	10	2	9
6	465	64	12	3	0
. . .					
2302	480	128	3	3	3
2303	488	98	4	7	2

Above matrix represents sample recover data for 48x48 watermark image. This 48x48 watermark image have 2304 pixels. Thus, watermark matrix has 2304 rows from 0 to 2303. The first column of watermark matrix is pixel number. The second and third columns represent Y and X coordinates of pixel in the original image. Next three columns of watermark matrix represent the square root of absolute magnitude value between original image and watermark image, which have respectively red (Wr(x,y)), green (Wg(x,y)), and blue (Wb(x,y)) color bands.

Generated recover data has been inserted into original image. This inserting process is called watermark embedding. The purpose of this embedding process, is to introduce a component call embedder to the novel proposed watermarking system. Inputs of embedder are original image and generated recover data from the previous steps. Process or task of embedder is to insert recover data into original image spatial using its embedding algorithms. Here it has denoted the original image as Fh, recover data as Fw, and resulting watermarked raw-image as Fwd. Equation 3.4 represents the relationship of original image, recover data, and watermarked raw-image.

$$Fwd = Fh + Fw$$

(3.4)

This study has introduced and developed three embedding algorithms for pixel based watermarking system. The first one is random insertion. The second one is insertion into less sensitive points to human vision. The third one is insertion into least significant bits (LSB) in JPEG macro-block.

3.2.1.1 Random Insertion

This method has not been considered the ordinate or position to insert recover data. It just inserts recover data (square root of absolute magnitude value between original image and watermark image) into original image at, the randomly selected points. Thus, embedding algorithm have properties of low time complexity, low capacity, and low memory requirements.

3.2.1.2 Insertion into Less Sensitive Pixel to Human Vision

This method is, just to insert recover data (square root of absolute magnitude value between original image and watermark image) into original image where, there is less sensitive pixel of original image. This is determined by less sensitive pixel based on the color value of each pixel of host image. This method is also very easy and fastest embedding method. It is very low time consuming and low computational complexity is provided. It provides good invisibility and keeps good capacity.

3.2.1.3 Insertion into Less Significant Point of JPEG Macro-block

This method inserts recover data (square root of absolute magnitude value between original image and watermark image) into original image which is, into LSB of the JPEG macro-block. It has been used 8x8 macro-block for this experiment. When encoding time, recover data will be no loss, because LSB pixels of the macro-block will not change. This method helps in easy recognition of watermark extraction process.

After inserting the recover data into original image, it results uncompressed watermarked image denoted as **Fwd**. This resulting object is extremely big and have much redundant data. Therefore, this raw-object cannot have distributed over the network or stored in storage media efficiently. It has to represent watermarked image in transferable and store-

able manner. Until now all work ware on uncompressed images in spatial domain. This means algorithms have manipulated each and every pixel individually. The purpose of efficient representation, is that it has to convert this raw-format watermarked object into encoded or compressed format. Thus, it has to apply the transformation method on the raw-watermarked object. That means now it has to convert raw-watermarked object from spatial domain to frequency domain. It was decided to use discrete cosine transformation known as DCT to encode the raw-watermarked object. Here is denoted the compressed watermark image as **Fwdc**. Similar encoding algorithm has been used for both pixel based and feature based watermarking methods. This chapter has described encoding algorithm in section `3.4 Encoder`.

3.2.2 Pixel Based Extraction

Watermark extracting processes should be adaptable and decoding algorithm should be simple. This study has developed a common watermark extraction algorithm for all above pixel based embedding methods. The first step of the extracting process is decompressing the watermarked image. It was decided to use traditional JPEG decompress algorithm to decode the watermarked image. Following this to extract the recover data from the watermarked image and reconstruct original or host image. Inputs of this extraction and recognition process are watermarked image and metadata file generated by embedding process. Figure 3.2 illustrates the high-level design diagram of pixel base watermark extraction technique.

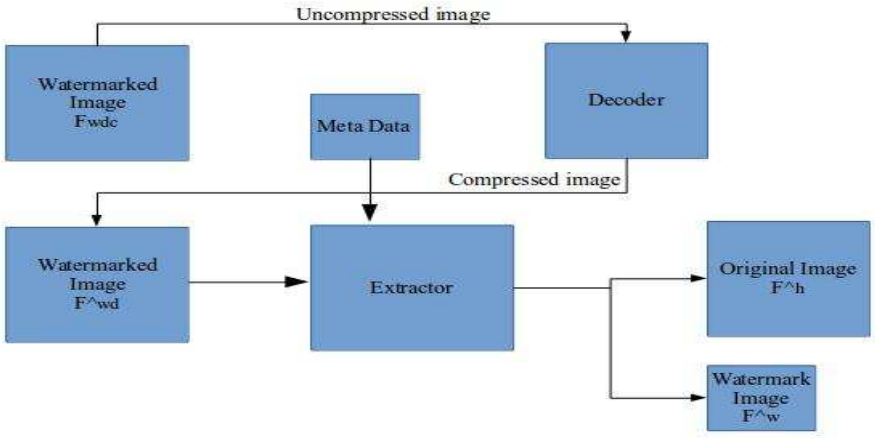


Figure 3.2: High level design diagram of pixel base watermark extraction technique

The watermark extraction process of the pixel based watermarking system has simple architecture. It has used minimum number of modules. Inputs of this extraction process are compressed watermarked images denoted by **F*wdc*** and metadata which is generated by the embedding process. The first step of extracting process is decompressing or decode the watermarked image. For this purpose, the DCT base decoder has been used. The similar decoding algorithm has been used for both pixel based and feature base watermarking methods. Decoding algorithm has been described in section `3.5 Decoder`. Output of the decoding process is decompressed watermarked image denoted by **F^*wd***. This object is approximately similar to the watermarked image previously mentioned in the embedding process denoted as **F*wd***. Because of JPEG compression uses quantization and it produces lossy-compressed image. Therefore, some information may have got los in the new decompressed image **F^*wd***.

Main module of the pixel based extracting process is an extractor. Inputs of extractor module are decompressed watermarked image **F^*wd*** and metadata which is generated by the embedding process. Extractor finds the recover data into watermarked image by using metadata. Metadata has been saved in matrix format, which is described in embedding section. In the pixel based methods, watermark matrix contains the spatial coordinates where, the watermark inserted. The outputs of the extractor are a watermark image **F^*w*** and image which is approximately similar to the original image.

3.3 Feature Based Techniques

This study has used low-level features of the host image such as edges and corners. This study, has used feature detection operation and steganography methods. The watermark embedding process is a challenging task, fidelity property and robustness property should be ensured. Edges and corners in the image are important features to represent the content of the image. The watermark embedding process has five sub processes: Identify edges and corners, generate recover data, decide regions which can be inserted recover data, add the recover data into host image, encoding watermarked image using a DCT base compression algorithm. Appendix A, Figure 2 illustrates the high-level design diagram of feature base watermarking techniques.

The digital image can be represented as a matrix of small elements called pixels. Each

pixel is represented by a numerical value from 0 to 255. In general, the pixel value is related to the brightness or color that will be seen when the digital image is converted into an analog image for display and viewing. In machine learning, pattern recognition and in image processing, feature extraction starts from an initial set of measured data and builds derived values (features) intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps, and in some cases leading to better human interpretations. Feature extraction is related to dimensionality reduction. Low level features of digital image have been used for invisible watermarking. Edges, corners, blob, and ridge are low level features, which are used in digital watermarking.

A corner can be defined as the intersection of two edges. A corner can also be defined as a point for which there are two dominant and different edge directions in a local neighborhood of the point. An interesting point is a point in an image which has a well-defined position and can be robustly detected. This means that an interest point can be a corner, but it can also be, for example, an isolated point of local intensity maximum or minimum, line endings, or a point on a curve where the curvature is locally maximal. In practice, most called corner detection methods detect interest points in general, and in fact, the term "corner" and "interest point" are used more or less interchangeably. As a consequence, if only corners are to be detected it is necessary to do a local analysis of detected interest points to determine which of these are real corners. Corner detectors are not usually very robust and often require large redundancies introduced to prevent the effect of individual errors from dominating the recognition task. One determination of the quality of a corner detector is its ability to detect the same corner in multiple similar images, under conditions of different lighting, translation, rotation and other transforms.

Inputs of the feature based watermarking system are original or host image and watermark. Original image is a compressed or uncompressed raw image. Normally original image is a large image. The watermark is a small compressed or uncompressed image. 48x48 color or grayscale image has been used as a watermark object. Figure 3.5 illustrates the high-level design diagram of pixel base watermark embedding technique.

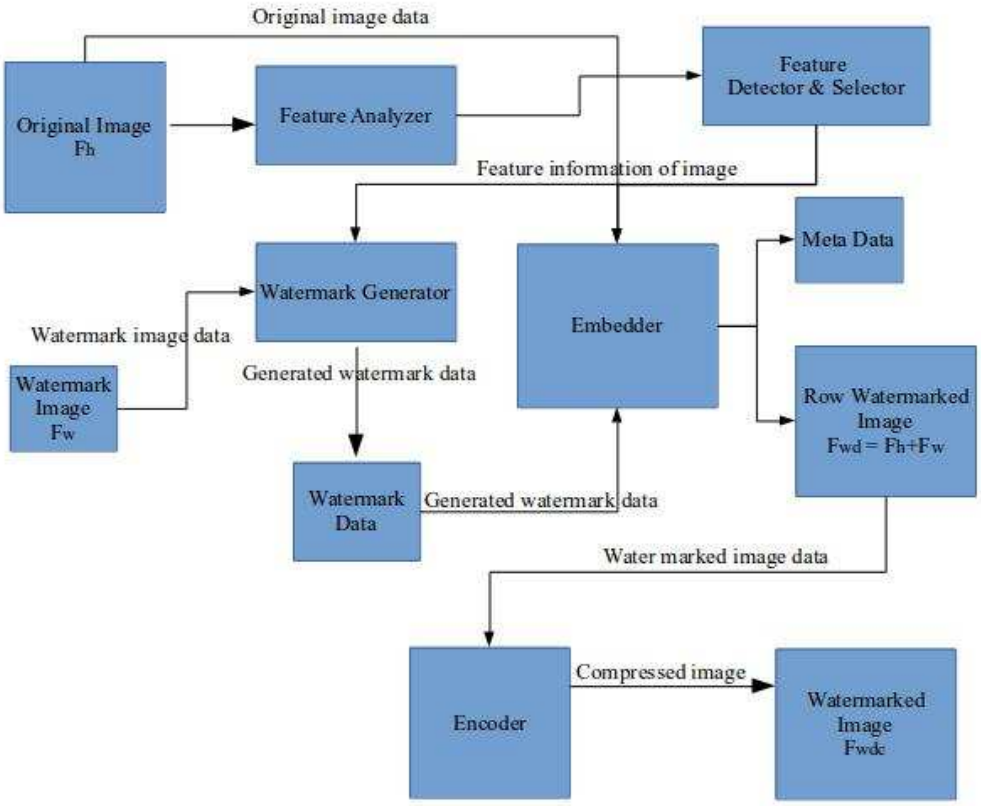


Figure 3.3: High level design diagram of feature base watermark embedding technique

3.3.1 Feature Detection

Original image is sens to feature analyzer and it analyses the low-level features of the original image. Feature analyzer sends the information about features of image to feature detector. Feature detector decides, what are the most prominent features and creates a binary image with features. This study has done experiments based on corners of digital image. Corner is a popular feature of images. Because of it is rotation invariant feature. Study has used two corner detection method in the feature detector module. The first one is popular Harris operator for corner detection. The second one is over novel introduced corner detection operator. Harris operator provides the rotation invariant feature detection properties. Novel operator has provided rotation invariant and scale invariant properties.

3.3.1.1 Feature Detection using Harris Operator

Chris Harris [35], a famous researcher in the image processing field, has done a comprehensive research and inverted very popular Harris corner detection method to the world. This operator is step by step process for finding the corners. The first step is finding the edges of horizontal and vertical directions. Then find the cross product of those edges. Then use the Gaussian operator to remove the noise. Finally, non-maximal suppression has done. Harris operator is not an individual operator. It's a step by step process as describes follows. Chapter 4 will describe the steps by step implementation of Harris operator in detail.

Step 1: Compute X and Y derivatives and compute products of derivatives using these matrices.

$$\partial X = [-1,0,1]$$
$$\partial Y = [-1,0,1]$$

(3.5)

Step 2: Apply Gaussian operator to Ix, Iy, Ixy.

Step 3: Create a matrix for each pixel(x,y) and compute response.

Step 4: Non-Maximum suppression of Harris response.

Harris operator can be described mathematically in this manner. Image given by ***I***, image patch over the area ***(u, v)*** and shifting it by ***(x, y)***. The weighted sum of squared differences between these two patches, denoted S, is given by equation 3.6.

$$S(x,y) = \sum\sum w\,(u,v), \big(I(u+x, v+y) - I(u,v)\big)^2$$

(3.6)

I(u+x,v+y) can be approximated by a Taylor expansion. ***Ix*** and ***Iy*** be the partial derivatives of ***I***. This approximation represents by equation 3.7.

$$I(u+x, v+y) \approx I(u,v) + I_x(u,v)x + I_y(u,v)y$$

(3.7)

The approximated weighted sum of squared differences between these two patches, denoted S, is given by equation 3.8.

$$S(x,y) \approx \sum\sum w\,(u,v), \left(I_x(u,v)x + I_y(u,v)y\right)^{2,}$$

(3.8)

Equation 3.9 represents the structure tensor. This matrix is a Harris matrix, and angle brackets denote averaging

$$A = \sum_u \sum_v w(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \langle I_x^2 \rangle & \langle I_x I_y \rangle \\ \langle I_x I_y \rangle & \langle I_y^2 \rangle \end{bmatrix}$$

(3.9)

Harris operator detects the true corners of the original image. Feature detector generates a grayscale corner image as output. This corner image will be one input of the embedder.

3.3.1.2 Feature Detection using Novel Operator

This study has introduced a novel feature detection operator by improving limitations of the traditional Harris operator. Harris operator provides only the rotation invariant feature detection. Novel operator has provided rotation invariant and scale invariant of both properties. This study has abstract a novel model by extending the traditional Harris model and Laplacian of Gaussian (LoG) operator. Novel proposed feature extraction operator shows an extreme ability to detect the true and immutable corners and it provides rotation invariant and scale invariant transformation properties. This novel operator is a step by

step process for detecting the corners. The first step is finding the edges of horizontal, vertical, and also diagonal directions. Then finding the cross product of those edges. Then use the Laplacian of Gaussian or Mexican Hat operator to blurring the image. Finally, non-maximal suppression has to be done. Novel operator is not an individual operator. It's a step by step process as follows. Chapter 4 will describe the steps by step implementation of the novel operator in detail.

The first step is finding the edges of horizontal, vertical, and also diagonal directions. For this purpose, it has used the Sobel operator. Sobel operator was introduced by Irwin Sobel and Gary Feldman, colleagues at the Stanford Artificial Intelligence Laboratory (SAIL). It is 3x3 first derivative image gradient operator. Sobel operator has implemented using 3x3 convolution. Convolutions **M*x*, M*y*, M*45*, M*135*** are calculated gradient difference X-direction, Y-direction, 45 angle and 135 angle respectively. Equation 3.10 represents the matrices of Sobel operator.

$$Mx = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} My = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

$$M45 = \begin{bmatrix} 0 & +1 & +2 \\ -1 & 0 & +1 \\ -2 & -1 & 0 \end{bmatrix} M135 = \begin{bmatrix} -2 & -2 & 0 \\ -1 & 0 & +1 \\ 0 & 1 & +2 \end{bmatrix}$$

(3.10)

The second step is finding the gradient magnitude and gradient directions of these edges. Equation 3.11 represents these calculations.

$$M = \sqrt{Mx^2 + My^2} \quad M = \sqrt{M45^2 + M135^2}$$

$$\emptyset = atan\left(\frac{\nabla My}{\nabla Mx}\right) \quad \emptyset = atan\left(\frac{\nabla M135}{\nabla M45}\right)$$

(3.11)

The third step uses the Laplacian of Gaussian or Mexican Hat operator to blur the image. The LoG is the negative normalized second derivative of a Gaussian function, i.e., up to

scale and normalization, the second Hermite function. It is a special case of the family of continuous wavelets (wavelets used in a continuous wavelet transform) known as Hermitian wavelets. The Ricker wavelet is frequently employed to model seismic data, and as a broad-spectrum source term in computational electrodynamics. It is usually referred to as the LoG wavelet in the Americas, due to its taking the shape of a sombrero when used as a 2D image processing kernel. Equation 3.12 illustrate the 2D LoG operator and Equation 3.13 illustrates the LoG kernel.

$$\psi(x,y) = \frac{1}{\pi\sigma^2}\left(1 - \frac{x^2+y^2}{2\sigma^2}\right)\mathrm{e}^{-(x^2+y^2)/2\sigma^2}.$$

(3.12)

$$\begin{pmatrix} 0 & 0 & 3 & 2 & 2 & 2 & 3 & 0 & 0 \\ 0 & 2 & 3 & 5 & 5 & 5 & 3 & 2 & 0 \\ 3 & 3 & 5 & 3 & 0 & 3 & 5 & 3 & 3 \\ 2 & 5 & 3 & -12 & -23 & -12 & 3 & 5 & 2 \\ 2 & 5 & 0 & -23 & -40 & -23 & 0 & 5 & 2 \\ 2 & 5 & 3 & -12 & -23 & -12 & 3 & 5 & 2 \\ 3 & 3 & 5 & 3 & 0 & 3 & 5 & 3 & 3 \\ 0 & 2 & 3 & 5 & 5 & 5 & 3 & 2 & 0 \\ 0 & 0 & 3 & 2 & 2 & 2 & 3 & 0 & 0 \end{pmatrix}$$

(3.13)

LoG function-based operator is used for image feature detection, including the local area detection and the feature point detection. For the feature point detection, the LoG operator is performed in scale space to get the key points. The LoG operator provides the best performance with precision and the detecting accuracy. Therefore, novel proposed method has improved the performance with the precision and the detection. Appendix A, Figure 3 illustrates the LoG or Mexican hat operator with different sigma values.

The fourth and final step is non-maximal suppression. The image is scanned along the image gradient direction, and if pixels are not part of the local maxima they are set to zero. This has the effect of suppressing all image information that is not part of local maxima.

Thus, non-maximum suppression can help suppress all the gradient values to 0 except the local maxima.

Novel corner detection operator has gotten the advantages from both the Sobel edge detector and Mexican Hat smoothing filter. The large convolution kernel of Sobel, smooths the input image to a greater extent and so makes the operator less sensitive to noise. The operator also generally produces considerably higher output values for similar edges, compared with the Harris gradient detector kernel. The output values of the Sobel operator can easily overflow the maximum allowed pixel value for image types that only support smallish integer pixel values. When this happens, the standard practice is to simply set overflowing output pixels to the maximum allowed value. The problem can be avoided by using an image type that supports pixel values with a larger range. Natural edges in images often lead to lines in the output image that are several pixels wide due to the smoothing effect of the Sobel operator. The LoG operator takes the second derivative of the image. Where the image is basically uniform, the LoG will give zero. Wherever a change occurs, the LoG will give a positive response on the darker side and a negative response on the lighter side. At a sharp edge between two regions, the response will be:

- zero away from the edge
- positive just to one side
- negative just to the other side
- zero at some point in between on the edge itself

LoG can be approximated by a Difference of two Gaussians (DoG) at different scales. Separability of and cascadability of Gaussians applies to the DoG, so can achieve efficient implementation of the LoG operator. DoG approx also explains bandpass filtering of LoG. LoG filter has improved scale invariant property of a novel watermarking system.

Harris operator and Novel corner detection operator both are implemented in a module called feature detector. Output signal of the feature analyzer, will be input of the feature detector module. Output of the feature detector is the binary image marked with detected corners.

3.3.2 Feature Based Watermark Generation

The feature based watermarking system also has a module watermark generator. Inputs of watermark generator module is a binary image which, is the output of the feature detector. The main difference of watermark generation in this feature base method, considers intensity values of corner points of the original image. In this case too, it has considered the separate color components of corner points or corner pixels. Thus, red, green, and blue components have been taken separately. **I*(x,y)*** denoted intensity value of pixel (x,y). **Ir*(x,y)***, **Ig*(x,y)***, **Ib*(x,y)*** are color component, which red, green, and blue color components respectively. Section 3.2.1 has described the color components of individual pixel.

The feature detector sends signals (binary image with corners) to the watermark generator. Another input of watermark generator is a watermark image. The watermark generator, generates the recover data according to watermark image and signal which is sent by feature detector. Recover data, calculate using corner information of original image and spatial information of watermark image. It has computed recover data using the difference between corner pixel intensity values of the original image and watermark image intensity values, finally taken the square root of absolute magnitude value between above difference. The formulas of calculating recover data, has been described in pixel base watermarking.

After calculating recover data to every color band, the result is to represent using recover data matrix. This watermark matrix is different from the previous method, because novel watermark matrix does not contain any spatial coordinates. Following matrix represent sample recover data for 48x48 watermark image. This 48x48 watermark image has 2304 pixels. Thus, watermark matrix has 2304 rows from 0 to 2303. The first column of watermark matrix is pixel number. Next three columns of watermark matrix represent the square root of absolute magnitude value between original image and watermark image, which is respectively red (**Wr*(x,y)***), green (**Wg*(x,y)***), and blue (**Wb*(x,y)***) color bands.

TABLE III. RECOVER DATA MATRIX OF FEATURE BASED METHODS

Pixel	Wr(x,y)	Wg(x,y)	Wb(x,y)
1	5	3	6
2	7	3	1
3	3	5	9
4	9	1	4
5	10	2	9
6	12	3	0
. . .			
2302	4	3	3
2303	7	7	2

3.3.3 Feature Based Embedding

Generated recover data has inserted into original image. This inserting process is called watermark embedding. The purpose of the embedding process in this method has been to introduce a component called embedder to the novel proposed watermarking system. Inputs of embedder are original image, binary image with corners, and generated recover data from the previous step. Process or task of embedder is to insert recover data into original image using embedding algorithms. This thesis has denoted original image as **F*h***, recover data as **F*w***, and resulting watermarked raw-image as **F*wd***. The equation has been described in the previous section which represent the relationship of original image, recover data, and watermarked raw-image. This research has introduced two embedding techniques to the feature based watermarking system. The first technique, inserts recover data around a single corner. The second technique, inserts recover data into the number of the corner points in entire image space.

3.3.3.1 Insert Around a Single Corner

Feature detector has selected a most prominent corner of the original image and embedder has inserted the recover data into pixels around the selected corner. Following 3.14 has

represented the original image in matrix form. Assume pixel p(2,2) as selected corner point. Embedder will insert recover data around pixel p(2,2) by creating sub matrix 48x48 which same size of the watermark image.

$$Fh = \begin{bmatrix} p(0,0) & p(1,0) & p(2,0) & p(3,0) & \dots & p(n,0) \\ p(0,1) & p(1,1) & p(2,1) & p(3,1) & \dots & p(n,1) \\ p(0,2) & p(1,2) & p(2,2) & p(3,2) & \dots & p(n,2) \\ p(0,3) & p(1,03) & p(2,3) & p(3,3) & \dots & p(n,3) \\ p(0,4) & p(1,4) & p(2,4) & p(3,4) & \dots & p(n,4) \\ p(0,5) & p(1,5) & p(2,5) & p(3,5) & \dots & p(n,5) \\ . & . & . & . & \dots & . \\ . & . & . & . & \dots & . \\ . & . & . & . & \dots & . \\ p(0,n) & p(1,n) & p(2,n) & p(3,n) & \dots & p(n,n) \end{bmatrix}$$

(3.14)

3.3.3.2 Insert into Multiple Corners

Feature detector has selected several corners in original image and embedder has inserted the recover data into pixels into selected corners. Above equalization has represented the original image in matrix form. Assume pixels p(2,0), p(3,1), p(1,5), p(3,5) are the selected corners. Embedder will insert recover data into all above corners.

After the recover data is inserted into the original image, the resulting uncompressed watermarked image is denoted as **Fwd**. This resulting object is extremely big and have much redundant data. Therefore, this raw-object is unable to distribute over the network or store in storage media in efficiency. It has to be represented watermarked image in a transferable and store-able manner. Until now algorithm has worked with uncompressed images in spatial domain. For the purpose of efficient representation, needs to convert this raw format watermarked object into encoded or compressed format. Thus, the transformation method has to applied on the raw-watermarked object. That means, it has to convert the raw-watermarked object from spatial domain to frequency domain. It has been decided to use discrete cosine transformation know as DCT to encode the raw-watermarked object. This thesis denoted compressed watermark image as **Fwdc**. A similar encoding algorithm has been used for both pixel based and feature base watermarking methods. The Encoding algorithms are described in section `3.4 Encoder`.

3.3.4 Feature Based Extraction

This study has introduced adaptive watermark extraction algorithm for the feature based method. The first step of extracting process is decompressing the watermarked image. It has decided to use traditional JPEG decompress algorithm to decode the watermarked image. Then it needs to extract the recover data from watermarked image and reconstruct original or host image. Inputs of this extraction and recognition process are watermarked image and metadata file generated by embedding process. Figure 3.7 illustrates the high-level design diagram of feature base watermark extraction technique.

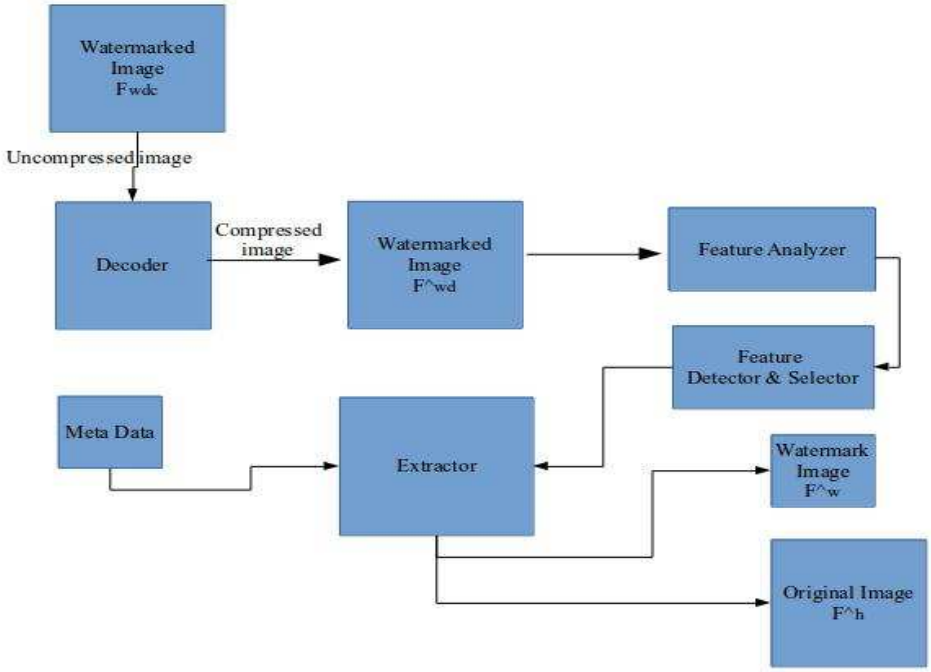


Figure 3.4: High level design diagram of pixel base watermark extraction technique

The watermark extraction process of the feature based watermarking system has a somewhat complex architecture compared to the previous pixel based method. This extraction process has modules decoder, feature analyzer, feature detector, and extractor. Inputs of this extraction process are compressed watermarked image denoted by **F*wdc*** and metadata which is generated by embedding process. The first step of the extracting process is to decompress or decode the watermarked image. For this purpose, DCT base decoder

has to be used. The decoding algorithms has been described in section `3.5 Decoder`. Output of the decoding process is decompressed watermarked image denoted by **F^*wd***. This object is approximately similar to the watermarked image previously mentioned in the embedding process denoted as **F*wd***..

Watermarked image is sent to the feature analyzer and it analyses the low-level features of the original image. Feature analyzer sends the information about features of image to feature detector. Feature detector generates a binary image with corners. This corner image is a input of extractor module. Main module of the pixel based extracting process is an extractor. Inputs of extractor module are decompressed watermarked image **F^*wd***, binary image with corners, and metadata which is generated by the embedding process. Extractor finds the recover data into watermarked image by using metadata. Metadata has been saved in matrix format which is described in the embedding section. Extra parameter is needed in this extraction process to select the embedding method. These parameters have passed via metadata information file. Outputs of the extractor are watermark image **F^*w*** which has been added in embedding process and image **F^*h*** which is approximately similar to the original image.

3.4 Encoder

This module does the encoding or compression. The previous module gives us raw-watermarked image as result. Before publishing or transmitting the raw-watermarked image it should be compressed into well-known image format such as JPEG. It was decided to used Discrete Cosine Transformation (DCT) and traditional JPEG compression with amendments.

This module converts the red, green, blue color channels to YCbCr space. After that, same proceeded carried out for a grayscale image is done to the Y, Cb, and Cr channels. Next, is to partition the image into blocks of size 8 x 8 pixels. Some more work is necessary if the dimensions of image are not divisible by 8. Then transform the image to the frequency domain using DCT. The DCT is a product of block(C), C = U B U^T where B is an 8 x 8 block from the preprocessed image and U is a special 8x8 matrix. DCT transformation tends to push most of the high intensity information in the 8x8 block to the upper left-hand of the block with the remaining values in C taking on relatively small values. The DCT is

applied to each 8x8 block.

The next step in the encoding module is the quantization. Here will make decisions about values in the transformed image elements near zero will converts to zero and other elements will be shrunk so that their values are closer to zero. All quantized values will then be rounded to integers.

The last step in the encoding sub process has removed the redundancy information of transformed and quantized image. In this case, it has followed regular JPEG standard uses, an advanced version of Huffman coding. Appendix A, Figure 4 illustrates the encoder module with detail.

3.5 Decoder

Above encoder modules are responsible to embed the recover data into the original image and create the watermarked image. This decoder module is responsible the extraction of the watermark from watermarked image. Watermark extracting process is adaptable and decoding algorithm is simple. To extract both directional and non-directional edge features, it needs to define small square image-blocks. This module divides an image space into non-overlapping square blocks and then it can extract edge information from each block.

The quantization process of JPEG compression results the lossy watermarked image. Extraction algorithm has gotten this lossy compressed image. But watermark information is hidden in the edges of the host image. Therefore, this lossy characteristics of compression process is not much effective to recovering data. Because of this, it was decided to use traditional JPEG decompress algorithm to decode the watermarked image. The decoding process, has followed the reverse run-length encoding and Inverse-DCT steps and the reconstructed bit map image.

3.6 Summary

This chapter has described in detail, methodology and design of novel process and what the system does. This chapter has described seven main modules, namely spatial analyzer, feature analyzer, feature detector, embedder, extractor, encoder, and decoder. This chapter

has discussed relationships and links between above modules. With a clear idea about what is done in the system, let's now move to the next chapter which describes, how experimental design and experimentation of each module is done and how the entire solution starts functioning.

Chapter 4

Experimental design and Experimentation

4.1 Introduction

The previous chapter gave detailed information about the methodology of the novel solution and what the system does. Chapter 3 describes the purpose of each and every module and link between those modules. This chapter describes the experimental design of the novel approach. This chapter describes how the experiments are done and how major modules work. Here it is described how prototypes are built using the help of material such as pseudo code and some main code segments of algorithms. It described at the beginning of the chapter, how to setup the experimental environment and background technologies have used in this research. Then it describes experimental design and experimentation regarding each module and entire system.

4.2 Implementation Support Technologies

Overall solution has been implemented as a Graphical User Interface (GUI) application, that can be user able to access on any platform such as Linux, Unix, and MS-Windows. It uses open-source technologies to develop the prototype application. Workable prototype has been developed using C/C++ programming language and GNU Compiler Collection (GCC). Libjpeg library was used to compress and decompress the image. Prototype implementation entirely based on open-source technologies.

C and C++ are high-level programming languages. C language is used for procedural programming. C++ has object-oriented and generic procedural programming features, while also providing facilities for low-level memory manipulation. It was designed to system programming and embedded, resource-constrained and large systems, with performance, efficiency and flexibility of use. C++ is standardized by the International Organization for Standardization (ISO), with the latest standard version ratified and published by ISO in December 2014 as ISO/IEC 14882:2014.

The GNU Compiler Collection (GCC) includes front ends for C, C++, Objective-C, Fortran, Ada, and Go, as well as libraries for these languages. GCC was originally written

as the compiler for the GNU operating system. The GNU system was developed to be 100% free software, free in the sense that it respects the user's freedom. GCC provides regular, high quality releases, which want to work well on a variety of native and cross targets (including GNU/Linux), and encourage every developer to contribute changes or help testing GCC. Source code of GCC fully accessible to read and freely available to download and use.

Libjpeg is a widely-used C library for reading and writing JPEG [25], [26] image files. It was developed by Tom Lane and the Independent JPEG Group (IJG) during the 1990's and it is now maintained by several community developers. The JPEG implementation of the Independent JPEG Group (IJG) was first publicly released on the 7th of October 1991, and has been considerably developed since that time. The latest release is version 6b of 27-Mar-1998. This is a stable and solid foundation for many application's JPEG support. Open Source software implementation of the IJG was one of the major Open Source packages, and was key to the success of the JPEG standard. Many organizations incorporated it into a variety of products such as image editors and web browsers.

OpenCV is an image processing library developed using the C++ programming language. OpenCV is released under a BSD license and hence it’s free for both academic and commercial use. It has C, C++, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with OpenCV, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform. Study has used OpenCV for the purpose of accelerating the developments and testing.

Qt is a Cross-platform development framework. It is more faster and smarter way to create innovative user interfaces for desktop applications and embedded devices. Qt applications can be run on various software and hardware platforms with little or no change in the underlying code base, while still being a native application with native capabilities and speed. Prototype has used Qt, purpose of the developing prototype implementation in GUI.

CMake is an open-source, cross-platform family of tools designed to build, test and package software. CMake is used to control the software compilation process using simple platform and compiler independent configuration files, and generate native make-files and workspaces that can be used in the compiler environment of your choice. The suite of CMake tools was created by Kit-ware in response to the need for a powerful, cross-platform build environment for open-source projects. Prototype has used CMake to build our prototype application.

4.3 Experimental Environment

Digital image processing is highly experimental base research area. Thus, digital images watermarking also highly experimental base. It was an essential requirement to setup a perfect experimental environment for the research. After a comprehensive review of the related technologies in literature, it was decided to select above technologies to build the experimental environment. It has built a prototype watermarking system for experimental testing of the novel proposed solution. This prototype application was a laboratory for the research.

Prototype system has decided to take advantage of powerful high level programming language such as C or C++. Most of image processing applications have been built using C/C++ language. Therefore, this prototype able to convert any commercial or noncommercial application easily. Another usage of C/C++ is, there are many supporting image processing libraries. Prototype application has used OpenCV which, is C++ based free and open-source library to support many platforms. It helps us for quick implementation and testing. The major output of the embedding process is a watermarked image. This watermarked image is compressed object by using a DCT base algorithm. Therefore, Prototype application has used libjpeg library for compressed raw-watermarked image. Prototype implementation is a GUI application. The purpose of developing the GUI application, it used Qt application development framework.

4.4 Dataset

In literature, it is found that a common dataset was used for image processing researches. This study also has used that dataset to do the experiment in this research. In Appendix B illustrate the dataset, used in this study. This dataset includes popular images were used in

image processing research widely Eg: lena images, monkey image vegetable image etc.

Literature has given the evidence; the watermarking systems have used a different type of watermark objects. This research has used 48x48 color or grayscale image as a watermark object. One salient feature of this watermarking system, it can use a color image as watermark object. Figure 4.1 illustrates the watermark objects.

Watermark 1 Watermark 2 Watermark 3 Watermark 4

Figure 4.1 Watermark objects

4.5 Experiments of Pixel Based Watermarking Techniques

Chapter 3, it is described pixel based watermarking techniques in detail. It explains the methodology and architecture of the pixel based watermarking system. This section presents the experiment of the novel pixel based solution. In this method, analysis the original image and watermark object and generate the recover data according to the original image and watermark object. That generated recover data insert into original image using an embedding algorithm and finally it encodes using compression algorithm based on discrete cosign transformation.

4.5.1 Watermark Generator of Pixel Based

The recover data generator is a one major module of the pixel based watermarking system. Inputs of watermark generator are output of spatial analyzer and recover data. The watermark generator generates the recover data according to watermark image and signal which sent by spatial analyzer. Recover data calculate using spatial information of original image and spatial information of watermark image. The recover data computes using difference between original image intensity values and watermark image intensity values. Finally, have taken square root of absolute magnitude value between original image and watermark image. Methodology chapter has described this module before. Implementation

of watermark generator module by using C programing language has included in appendix D. The output of this module is recover data matrix. This also has described in methodology chapter. This watermark matrix practically saves as csv file in prototype implementation.

4.5.2 Watermark Embedder of Pixel Based

Watermark embedder is one major module of pixel based watermarking system. Inputs of embedder are original image and recover data. In pixel based method watermark embedding process, go through each and every pixel of host image and embed watermark image data into those pixels. This embedding process, has added recover data into original image in spatially. Prototype application has developed three techniques and those are experimentally tested. The first one is randomly insertion. The second one is inserted into less sensitive points to human vision. The third one is inserted into least significant point in JPEG macro-block. Methodology chapter has described this module before. Implementation of watermark embedder module by using C programing language has included in appendix D. Outputs of an embedder module are watermarked image in raw-format and metadata file in csv which containing recover data matrix.

4.5.3 Encode in Pixel Based

Encoder module does the encoding or compression. Previous module gives us raw watermarked image as a result. Before publishing or transmitting, raw-watermarked image should be compressed to well-known image format such as JPEG. This study has used Discrete Cosine Transformation (DCT) and traditional JPEG compression with amendments. DCT transformation tends to push most of the high intensity information in the 8x8 block to the upper left-hand of the block. The DCT is applied to each 8x8 block. The next step in the encoding module is the quantization. The last step in the encoding process has removed the redundancy information of transformed and quantized image.

This study has done a number of experimental test cases in the pixel based watermarking system. These test cases gave the evidence to us about success of the research. Table IV represents embed using random insertion method of the pixel base watermarking system. Appendix B contains the complete experimental results of pixel based watermark embedding methods.

TABLE IV. EXPERIMENT 1: EMBEDDING USING RANDOM INSERTION

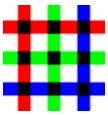
Inputs		Outputs	
Base method	: Pixel base	Watermarked image	: result_lena.jpg
Technique	: Random insertion	Metadata file	: result_lena.csv
Original image	: lena.jpg		
Watermark	: Watermark 4		

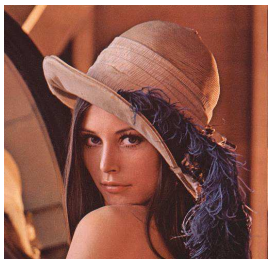
Compression parameters:

Block size: 8x8

Quality: 70%

JSAMPROW row pointer size: 1





Original image	Watermark image	Watermarked image
Dimension: 512x512	Dimension: 48x48	Dimension: 512x512
Size: 37.9 Kb	Size: 22.8 Kb	Size: 45.4 Kb

4.5.4 Watermark Extraction of Pixel Based

The watermark extraction process of the pixel based watermarking system has simple architecture and this has described this in methodology chapter. Here it is implemented using minimum number of modules. Inputs of this extraction process are compressed watermarked image and metadata which is generated by embedding process. The first step of extracting process is decompressing or decode the watermarked image. For this purpose, DCT base decoder has used to decoding algorithm, which in Chapter 3. Output of decoding process is decompressed watermarked image. This image approximately similar to the watermarked image previously mentioned at embedding process.

Main module of the pixel based extracting process is an extractor. Inputs of extractor module are decompressed watermarked image and metadata which is generated by embedding process. Extractor finds the recover data into watermarked image by using metadata csv file. In this method, recover data file contains the spatial coordinates where,

the watermark inserted. Outputs of extractor is watermark image, where has added in embedding process and the image, which approximately similar to the original image.

Study has developed a common extraction algorithm for the pixel based watermarking system. Thus, able to be applied to this common algorithm independently embedding technique. Nothing to worry about, what embedding technique has used to insert the watermark. Thus, pixel based extraction don't want ~~to~~ extra parameter to decide the embedded technique of the watermark. Table V is represented extraction processes of the pixel base watermarking system. In Appendix B contain the complete experimental results of the pixel based watermark extraction methods.

TABLE V. EXPERIMENT 2: EXTRACTION USING COMMON ALGORITHM

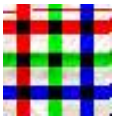
Inputs	**Outputs**
Base method : Pixel base Technique : common Original image : result_lena.jpg Metadata file : result_lena.csv	Extracted watermark: wtm.jpg

Compression parameters:

Block size: 8x8

Quality: 70%

JSAMPROW row pointer size: 1



Watermarked image Dimension: 512x512 Size: 45.4 Kb	Extracted watermark Dimension: 48x48 Size: 1.9 Kb

4.6 Experiments of Feature Based Watermarking Techniques

Chapter 3, describes feature based watermarking techniques in detail. This section presents the experimental design and experimentation of the novel feature based solution.

4.6.1 Feature Analyzer, Feature Detector & Watermark Generator

Feature based watermarking system has major modules called feature analyzer, feature detector, and watermark generator. Input of feature analyzer and feature detector is the original image. Output of feature detector is a binary image with corner information. Inputs of watermark generator module is a binary image and watermark object. The main difference of watermark generation in this feature base method, this consider intensity values of corner points of the original image. Original image send to feature analyzer and it analysis the low-level features of the original image. Feature analyzer send the information about features of image to feature detector. Feature detector decide, what are the most prominent features and create binary image with features.

This study has done research based on corners of digital image. This method has used two corner detection method in the feature detector module. The first one is popular Harris operator for corner detection. The second one is over novel introduced corner detection operator. Harris operator provides the rotation invariant feature detection properties. Novel operator has provided rotation invariant and scale invariant both properties. Methodology chapter has described these modules before. Implementation of feature analyzer, feature detector, and watermark generator modules by using C programing language has included in appendix D. The outputs of watermark generator module is recover data matrix. It has described it methodology chapter. This watermark matrix practically saves as csv file in this prototype implementation.

4.6.2 Watermark Embedder of Feature Based Method

Generated recover data have inserted into original image. The purpose of this embedding process, used module called embedder, it has described in Chapter 3. Inputs of embedder are original image, binary image with corners, and generated recover data from the previous step. The process or task of embedder is insert recover data into original image using embedding algorithms. Here it is described experimental test cases of the feature based embedding. Implementation of feature based watermark embedder modules by using C programing language has included in appendix D.

This research has done experiments of two embedding techniques to feature based

watermarking system. The first technique, insert recover data into around the single corner. The second technique, insert recover data into number of corner points in entire image space.

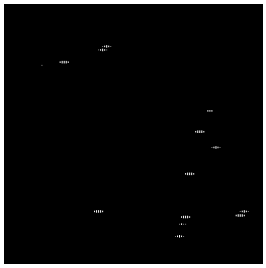
This study has done several experimental test cases in feature based watermarking system. These test cases gave the evidence to us about success of the novel corner detection method and embedding method based on low level features. This thesis represents experimental results separately by Harris operator and novel introducing corner detector operator.

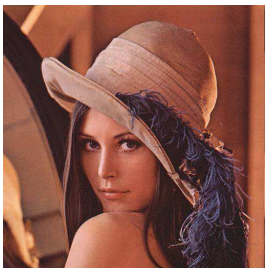
4.6.2.1 Experiment using Harris Operator

There are large number of experiments using Harris corner detector by changing parameters Harris operator. Harris operator have three parameters namely kernel, sigma, and threshold which can change in prototype system. Embedder module has a parameter, embedding area which can change in prototype system. In Appendix B contain the full experimental results of Harris based watermark embedding methods. Sample experimental results have represented in Table VI.

TABLE VI. EXPERIMENT 3: EMBED USING THE HARRIS CORNER DETECTOR

Inputs	Base method: Feature base Position: Around single corner Original image: lena.jpg Watermark: Watermark 4
Outputs	Watermarked image: result_lena.jpg Metadata file: result_lena.csv
Settings	Feature detector: Harris operator Smoothing: Gaussian filter Kernel: 5x5 Sigma: 1 Threshold: 120
Compression parameters	Block size: 8x8 Quality: 70% JSAMPROW row pointer size: 1





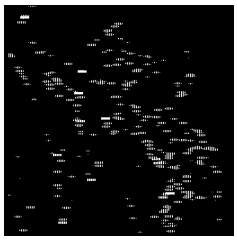
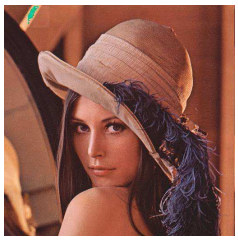
Original image Dimension: 512x512 Size:37.9 Kb	Binary corner image Dimension: 512x512 Size: 26.2 Kb	Watermarked image Dimension: 512x512 Size: 46.3 Kb

4.6.2.2 Experiment using Novel Introduced Operator

There are large number of experiments using Novel proposed corner detector by changing parameters. Novel corner detection operator has three parameters, namely kernel, sigma, and threshold which can change in this prototype system. Embedder module has a parameter, embedding area which can change in prototype system. In Appendix B contain the complete experimental results of a Novel corner detector based watermark embedding. Sample experimental results have represented in Table VII.

TABLE VII. EXPERIMENT 4: EMBED USING NOVEL CORNER DETECTOR

Inputs	Base method: Feature base Position: Around single corner Original image: lena.jpg Watermark: Watermark 4
Outputs	Watermarked image: result_lena.jpg Metadata file: result_lena.csv
Settings	Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 120
Compression parameters	Block size: 8x8 Quality: 70% JSAMPROW row pointer size: 1

		
Original image Dimension: 512x512 Size: 37.9 Kb	Binary corner image Dimension: 512x512 Size: 19.3 Kb	Watermarked image Dimension: 512x512 Size: 47.2 Kb

4.6.3 Watermark Extraction of Feature Based Method

The watermark extraction process of the feature based watermarking system has a bit complex architecture. Methodology chapter has described this before. Here has described experimentations and results of feature based extraction here. Major modules of an extraction processes are decoder, feature analyzer, feature detector, and extractor. Inputs of this extraction process are compressed watermarked image and metadata which is generated by embedding process. The first step of extracting process is decompressing or decode the watermarked image. For this purpose, here has used DCT base decoder. Thesis has described decoding algorithm in Chapter 3. Output of the decoding process is decompressed watermarked image. This object is approximately similar to the watermarked image previously mentioned at embedding process and not extremely.

Watermarked image sends to feature analyzer and it analysis the low-level features of the original image. Feature analyzer sends the information about features of image to feature detector. Feature detector generates a binary image with corners. This corner image is a one input of module extractor. Main module of the pixel based extracting process is an extractor. Inputs of extractor module are decompressed watermarked image, binary image with corners, and metadata which is generated by embedding process. Extractor finds the recover data into watermarked image by using metadata. Metadata has saved in matrix format which described in embedding section. Extra parameter need to this extraction processes to select the embedding method. Practically passing these parameters via a metadata information file. Outputs of the extractor are watermark image, has added in embedding process and image which, approximately similar to the original image. In Appendix B contain the complete experimental results of feature base watermark

extraction. Sample experimental results of Harris method has represented in Table VIII and sample experimental results of novel method has represented in Table IX.

TABLE VIII. EXPERIMENT 5: EXTRACT USING HARRIS CORNER DETECTOR

Inputs	Base method: Feature base Position: Around single corner Original image: lena.jpg Watermark: Watermark 4
Outputs	Watermarked image: result_lena.jpg Metadata file: result_lena.csv
Settings	Feature detector: Harris operator Smoothing: Gaussian filter Kernel: 5x5 Sigma: 1 Threshold: 120/130
Compression parameters	Block size: 8x8 Quality: 70% JSAMPROW row pointer size: 1

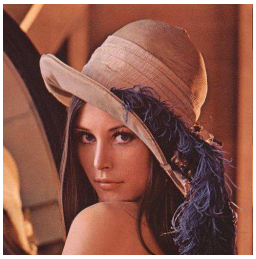
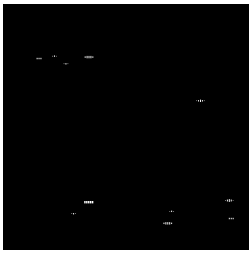
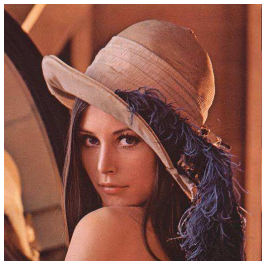
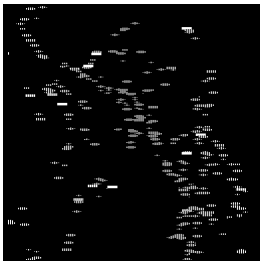
		
Original image Dimension: 512x512 Size: 94.4 Kb	Binary corner image Dimension: 512x512 Size: 26.2 Kb	Extracted watermark Dimension: 48x48 Size: 1.6 Kb

TABLE IX. EXPERIMENT 6: EXTRACT USING NOVEL CORNER DETECTOR

Inputs	Base method: Feature base Position: Around single corner Original image: lena.jpg Watermark: Watermark 4
Outputs	Watermarked image: result_lena.jpg Metadata file: result_lena.csv

Settings	Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 120/130
Compression parameters	Block size: 8x8 Quality: 70% JSAMPROW row pointer size: 1





Original image Dimension: 512x512 Size: 94.4 Kb	Binary corner image Dimension: 512x512 Size: 46.2 Kb	Extracted watermark Dimension: 48x48 Size: 1.6 Kb

4.7 Summary

This chapter described experimental design of the novel proposed watermarking system and presented experimentations and experimental results of each embed and extraction methods. This chapter gives evidence of successful implementation of experimental environment. With a clear Idea about experimental results, then move to next chapter which describes evaluation of experimental results and how much efficient the novel approach proposes in this thesis.

Chapter 5

Evaluation of Novel Watermarking Approach

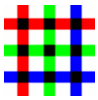
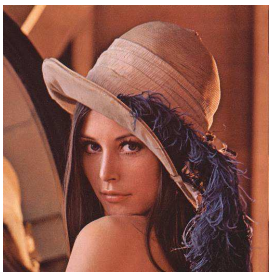
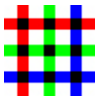
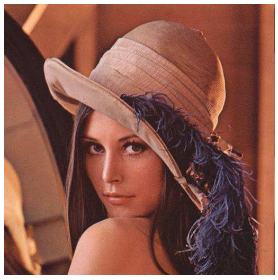
5.1 Introduction

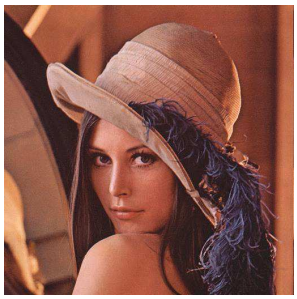
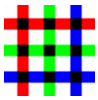
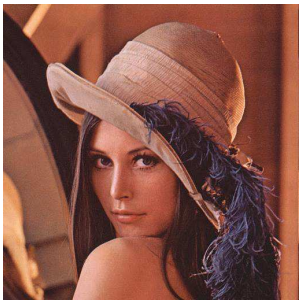
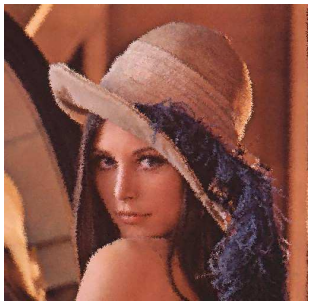
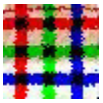
Chapter 4 deals with experimental design and experimentation of this study. This chapter described the evaluation of experimental results. This chapter has carried out evaluation according to the main characteristics of digital watermarking. Major characteristic of the watermarking algorithms, namely robustness, fidelity, and capacity have been considered. Experimental and mathematical methods have been used in the evaluation process. This evaluation process has used common dataset, widely used in image processing researches.

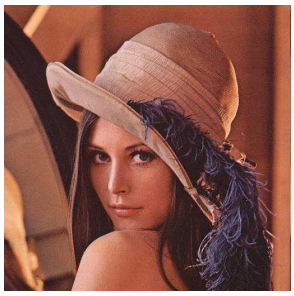
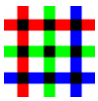
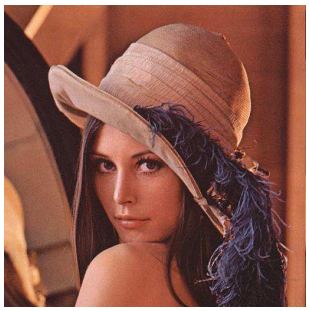
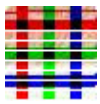
5.2 Evaluate of the Robustness

This section, has evaluated the robustness of the watermarking system. Feature based watermarking methods, using Harris operator and novel corner detection operator have been evaluated. Many types of attacks have been identified in the literature. This study has examined the evaluation against commonly identified attack type noise adding, filtering, rotation, and scaling. Appendix C contains the full evaluation results of the novel watermarking algorithm. Sample results of robustness evaluation has been represented in Table X.

TABLE X. SUMMARY OF EVALUATION RESULTS FOR THE ROBUSTNESS

Settings & Attack	Watermark	Watermarked Image	Watermarked Image After Attack
Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner **HSV noise** [Holdness: 3] [Hue:72] [Saturation:146] [Value: 94]		 	 
Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner **HSV noise** [Holdness: 3] [Hue:72] [Saturation:146] [Value: 94]		 	 

Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1.5 Threshold: 125 Position: single corner **Random Pik.** [Random seeds: 1452988117] [Randomization: 42] [Repeat: 5]		 	 
Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1.5 Threshold: 125 Position: single corner **Random Pik.** [Random seeds: 1452988117] [Randomization: 42] [Repeat: 5]		 	 

Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1.5 Threshold: 125 Position: single corner **Rotation in 45 degree**		 	 
Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1.5 Threshold: 125 Position: single corner **Scaling** watermarked image: 512x512 scaled to: 256x256		 	 

The above table has described evaluation methods and results related to the robustness of the watermarking algorithms. The first column of the table has represented settings, parameters, and attack types. The second column has represented watermark objects. The third and fourth columns represent evaluation results by experimentally. These columns show extracted watermark before and after the attack.

Evaluation results in table 5.1 and robustness evaluation table in Appendix C have giving evidence, novel watermarking method provides good surveillance against several types of noise adding attacks. Especially novel watermarking method provides rotation-invariant and scale-invariant characteristics both. Above tables has represented the results against rotation attack and scaling attack.

5.3 Evaluate of the Fidelity

This section, has evaluated, how far achieved fidelity property this novel watermarking system. This study has evaluated feature based watermarking methods which, using Harris operator and novel corner detection operator. Degradation of the original image information is natural after applied any watermark embedding algorithm. How compare the information degradation between original image and watermarked image? It is an important question. Various statistical methods have used in literature, for evaluating this degradation. Mean Square Error (MSE), Peak Signal to Noise Ratio (PSNR), and Structural Similarity Index Matrix (SSIM). Most of the watermarking researchers have used above methods to evaluate their research.

In statistics, the MSE of an estimator measures the average of the squares of the errors or deviations that is, the difference between the estimator and what is estimated. MSE is a risk function, corresponding to the expected value of the squared error loss or quadratic loss. The difference occurs because of randomness or because the estimator doesn't account for information that could produce a more accurate estimate. If **Y**^ is a vector of **n** prediction, and **Y** is the vector of observed values corresponding to the inputs to the function which generated the predictions, then the MSE of the predictor can be estimated by:

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

(5.1)

Prototype application has developed an algorithm to implement the above equalization in the system. Thus, it can measure the MSE value using watermarking system. According to the implementation, the return value can be 0 to 900 (If MSE<=1e-10, return will be 0). If the return value is lower, the best fidelity or higher similarity is assured.

Peak signal-to-noise ratio, often abbreviated PSNR, is an engineering term for the ratio between the maximum possible power of a signal and the power of corrupting noise that

affects the fidelity of its representation. Because many signals have a very wide dynamic range, PSNR is usually expressed in terms of the logarithmic decibel scale. PSNR is most commonly used to measure the quality of reconstruction of lossy compression codecs (e.g., JPEG image compression). The signal in this case is the original data, and the noise is the error introduced by compression. When comparing compression codecs, PSNR is an approximation to human perception of reconstruction quality. Although a higher PSNR generally indicates that the reconstruction is of higher quality, in some cases it may not. One has to be extremely careful with the range of validity of this metric; it is only conclusively valid when it is used to compare results from the same codec and same content. Mathematical representation of PSNR is:

$$PSNR = 10.log_{10}\left(\frac{MAX_i^2}{MSE}\right)$$

$$PSNR = 10.log_{10}\left(\frac{MAX_i^2}{MSE}\right)$$

$$PSNR = 20.log_{10}MAX_i - 10.log_{10}MSE$$

(5.2)

Prototype application has developed algorithm to implement above equalization in prototype system. Thus, it can measure the PSNR value using prototype watermarking system. According to the implementation return value can be 30 to 50. If return value is higher, the best fidelity or higher similarity is assured.

The structural similarity index is a method for predicting the perceived quality of digital television and cinematic pictures, as well as other kinds of digital images and videos. An early variant was developed in the Laboratory for Image and Video Engineering at The University of Texas at Austin and the full algorithm was developed jointly with the Laboratory for Computational Vision at New York University. The SSIM is used for measuring the similarity between two images. The SSIM index is a full reference metric. In other words, the measurement or prediction of image quality is based on an initial uncompressed or distortion-free image as reference. SSIM is designed to improve on traditional methods such as peak signal-to-noise ratio and mean squared error, which have proven to be inconsistent with human visual perception.

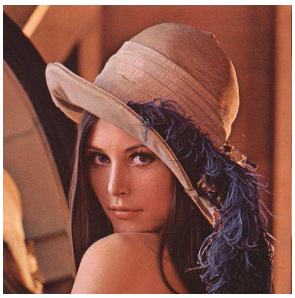
The SSIM index is computed on various windows of an image. The measure between two windows u* and u^0 of common size N×N is:

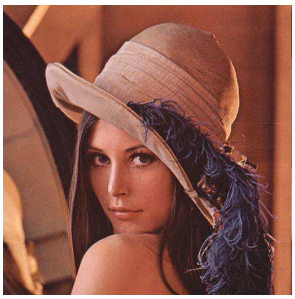
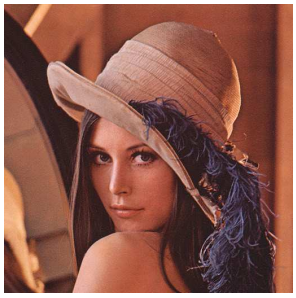
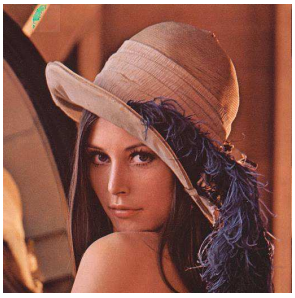
$$\mathrm{SSIM}(u^*,u^0)=\frac{(2\mu_{u^*}\mu_{u^0}+c_1)(2\sigma_{u^*u^0}+c_2)}{(\mu_{u^*}^2+\mu_{u^0}^2+c_1)(\sigma_{u^*}^2+\sigma_{u^0}^2+c_2)}$$

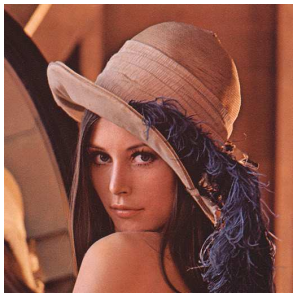
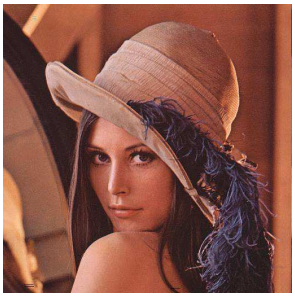
(5.3)

Prototype application has developed algorithm by supporting OpenCV, purpose of implementing above equalization in prototype system. This algorithm returns values Chanel 0, Chanel 1, Chanel 2, and mean. Appendix C contains the full evaluation results of the novel watermarking algorithm. Sample results of fidelity evaluation has been represented in Table XI.

TABLE XI. SUMMARY OF EVALUATION RESULTS FOR THE FIDELITY

Image Detail & Settings	Evaluation Results	Original Image	Watermarked Image
Image: lena.jpg (512x512, 94.4 Kb) **Settings** Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner	**Evaluation Method 1: MSE** Resulting Value: 102.03 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 39.28 Conclusion: Good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel [0]: 0.69 Chanel [1]: 0.75 Chanel [2]: 0.74 Mean: 0.72		

	Conclusion: Good Fidelity		
Image: lena.jpg (512x512, 94.4 Kb) **Settings** Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	**Evaluation Method 1: MSE** Resulting Value: 113.03 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 41.28 Conclusion: Good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel [0]: 0.69 Chanel [1]: 0.75 Chanel [2]: 0.74 Mean: 0.72 Conclusion: Good Fidelity		
Image: lena.jpg (512x512, 94.4 Kb) **Settings** Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner	**Evaluation Method 1: MSE** Resulting Value: 76.03 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 44.36 Conclusion: Very good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel [0]: 0.79 Chanel [1]: 0.85 Chanel [2]: 0.84 Mean: 0.82		

	Conclusion: Very good Fidelity		
Image: lena.jpg (512x512, 94.4 Kb) **Settings** Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	**Evaluation Method 1: MSE** Resulting Value: 54.44 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 45.51 Conclusion: Very good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel [0]: 0.79 Chanel [1]: 0.85 Chanel [2]: 0.84 Mean: 0.82 Conclusion: Very good Fidelity		

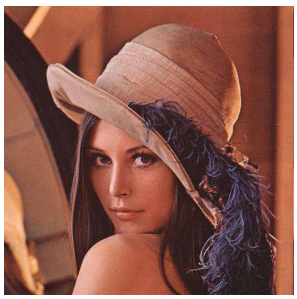
Above table has described evaluation methods and results related to fidelity between original image and watermarked image generated by embedding algorithm. The first column of the table represents input details, settings, and parameters of embedding process. The second column represents statistical evaluation results. This column shows mean square error, peak signal to noise ratio, structural similarity index, and conclusion according to those measurements. The third and fourth columns have represented evaluation results through experiments.

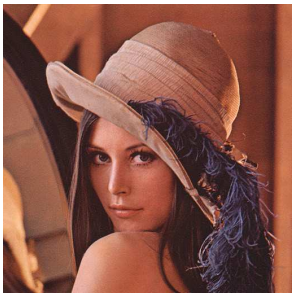
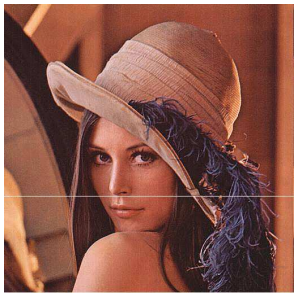
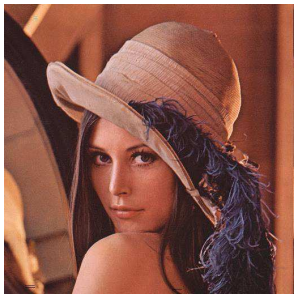
5.4 Evaluate of the Capacity

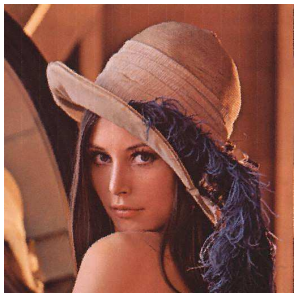
Capacity describe the number of bits of recover data or signal able to insert into original image or object with minimum destruction. In other words, Capacity is defined by using the largest quantity of information that inserted watermarks are capable of hiding. The

number of bits that can be inserted through watermarking varies with each application. Embedding algorithm decides number of bits or size of watermark that can be inserted into the original object. Expectation of watermarking system, hidden watermark information should not be destroyed or damaged. Therefore, in order to improve the security of the algorithm can enlarge the embedded space, and increase the size of number of bits into small pieces of the cover image. Fidelity property also relates to capacity. Because if the number of recover data or bits inserted into host image increased, it's difficult to hide from the observer. During this study, Experiments and evaluation of the embedding algorithm using different sizes of watermark objects has been done during this study. In this study, 48x48, 64x64, and 128x128 size watermark images have been used. Appendix C contains the full evaluation results of the capacity of the novel watermarking algorithm. Sample evaluation results of capacity has been represented in Table XII.

TABLE XII. SUMMARY OF EVALUATION RESULTS FOR THE CAPACITY

Image Detail & Settings	**Evaluation Results (Statistically)**	**Evaluation Results (Experimentally)** ***Original Image***	***Evaluation Results (Experimentally)*** ***Watermarked Image***
Image: lena.jpg (512x512, 94.4 Kb) **Watermark object:** Name: watermark 4.jpg Dimension: 48x48 px, Size: 22.8 Kb **Settings** Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	**Evaluation Method 1: MSE** Resulting Value: 103.03 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 41.28 Conclusion: Good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel [0]: 0.69 Chanel [1]: 0.75 Chanel [2]: 0.74 Mean: 0.72		

	Conclusion: Good Fidelity		
Image: lena.jpg (512x512, 94.4 Kb) **Watermark object:** Name: watermark 4.jpg Dimension: 128x128 px, Size: 29.8 Kb **Settings** Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	**Evaluation Method 1: MSE** Resulting Value: 344.05 Conclusion: Very bad fidelity **Evaluation Method 2: PSNR** Resulting Value: 23.82 Conclusion: Very bad fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel [0]: 0.59 Chanel [1]: 0.64 Chanel [2]: 0.66 Mean: 0.63 Conclusion: Bad Fidelity		
Image: lena.jpg (512x512, 94.4 Kb) **Watermark object:** Name: watermark 4.jpg Dimension: 48x48 px, Size: 22.8 Kb **Settings** Feature detector: Novel operator Smoothing: LoG Kernel: 5x5	**Evaluation Method 1: MSE** Resulting Value: 54.44 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 45.51 Conclusion: Very good fidelity **Evaluation Method 3: SSIM** Resulting Value:		

Sigma: 1 Threshold: 125 Position: multiple corners	Chanel [0]: 0.79 Chanel [1]: 0.85 Chanel [2]: 0.84 Mean: 0.82 Conclusion: Very good Fidelity		
Image: lena.jpg (512x512, 94.4 Kb) **Watermark object:** Name: watermark 4.jpg Dimension: 128x128 px, Size: 27.6 Kb **Settings** Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	**Evaluation Method 1: MSE** Resulting Value: 163.04 Conclusion: Bad fidelity **Evaluation Method 2: PSNR** Resulting Value: 40.04 Conclusion: Fair fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel [0]: 0.60 Chanel [1]: 0.65 Chanel [2]: 0.66 Mean: 0.64 Conclusion: Fair Fidelity		

The above table has described evaluation methods and results related to the capacity of watermarked image generated by the embedding algorithm. The first column of the table has represented input details, settings, and parameters of the embedding process. The second column represents statistical evaluation results. This column shows relationship of capacity and fidelity. The third and fourth columns have represented evaluation results through experiments. These columns show the relationship between human vision and capacity.

5.5 Summary

This chapter has presented an evaluation of the novel digital image watermarking system. The evaluation of all the algorithms, by world approved standards has been represented. The evaluation process has followed two paths. The first one is mathematical evaluation. The second method is experimental base evaluation. Evaluation of robustness, has been done by experimental base methods. Evaluation of fidelity and capacity have been done by using mathematical and experimental methods. After critical analysis, the results of the evaluation, are written in the conclusion in the next chapter.

Chapter 6

Conclusion

6.1 Introduction

Chapter 5 has presented a critical evaluation of the research process. This chapter of the thesis, describes conclusions of research findings, according to the results of critical evaluation. Until this point the thesis has collected the materials, dataset and results to check the hypothesis. This chapter, presents a critical review of research work against the experimental results. Finally, the conclusions of the research have been given.

The aim of invisible watermark is improving the authorization and protect the copyrights of real authors. Research gap in this study is finding the areas of inadequate attention given to the digital watermarking techniques and find the solution for secure authorship of digital image. Hypothesis of the study is mentioned problem which, can be solved by introducing novel feature base digital watermark embedding method and adaptive watermark extraction method. At the introduction, were defined three major objectives of the research. The first one is developing the robust invisible watermarking technique for digital image. The second objective is developing adaptive watermark extraction technique. The third objective is to evaluating the robustness, fidelity, capacity, and adaptability of novel embedding and extraction methods.

In this research process, a large amount of experiments has been done. This research has followed an extremely scientific method of research and abstracted a novel model of digital image watermarking. Proposed model is based on low-level features of the digital images. Background mathematical theories that have been used in novel model are first order derivative operator known as Sobel, second order derivative operator known as Laplacian of Gaussian, and Discrete Cosine Transformation. Novel proposed model has six major modules namely, feature detector, watermark generator, watermark embedder, watermark extractor, encoder, and decoder.

6.2 Major Findings

The literature [35]–[37], [39], has identified low-level features of digital images which has

used for digital watermarking. It was decided to used most salient feature of image called corners. Methodology chapter, has introduced a novel corner detection technique. Proposed corner detection operator is a step by step process. Study has used first order derivative operator and second order derivative both inside this operator. It also has described this novel operator with detail in methodology chapter. This novel operator has used recover data generation, recover data embedding and extraction in watermarking system. This operator is the most prominent part of entire research. The novel corner detection operator has gotten the advantages from both the Sobel edge detector and Mexican Hat smoothing filter. The large convolution kernel of Sobel, smoothens the input image to a greater extent and so makes the operator less sensitive to noise. The operator also generally produces considerably higher output values for similar edges, compared with the Harris gradient detector kernel. LoG can be approximated by a Difference of two Gaussians (DoG) at different scales. Separability of and cascadability of Gaussians applies to the DoG, so it can achieve efficient implementation of the LoG operator. DoG approx also explains bandpass filtering of LoG. LoG filter has improved scale invariant property of a novel watermarking system. During this study, a large number of experiments has been done by using proposed operator. All those experiments have been described in experimental design chapter and Appendix B. Experiments have been giving evidence to the success of this novel feature detector operator in digital image watermarking. Thesis has described the evaluation of the novel operator in evaluation chapter. It has presented the evaluation results with different parameters of Harris operator and novel proposed operator. The valuation results have proven better robustness and fidelity provided by the novel feature detection operator.

Watermark generator is another major module of watermarking system. Recover data generation process in this thesis is a dynamic one. Recover data, generates according to features of host image and intensity values of watermark object. This process has also used feature detection operator and it has described in methodology chapter. Experimental design and experimentation chapter and appendix D have described implementation of this module.

Embedder is a major module of the entire watermarking system. In feature based methods, low-level features of image have been used to insert recover data into host image. An

embedding process has also used feature detector operator to decide the embedding points of recover data. This thesis introduces two methods to insert recover data into host image. The first method is to embedding around single corner points. The second one has embedded recover data into more than one corner points. These methods have described in Chapter 3, and experimental details have been described in Chapter 4. Results of critical evaluation of embedding methods have been described in Chapter 5. Mathematical and experimental evaluation has given evidence for success of the novel embedding algorithm.

Chapter 5 has done evaluation according to main characteristic of digital watermarking. This study has considered major characteristics of watermarking algorithms, namely robustness, fidelity, and capacity. Experimental and mathematical methods have been used in this evaluation process. The experimental design has described separate test cases for each characteristic and conducted a separate evaluation for each and every characteristic that should be present watermarking system. The evaluation results of robustness have been giving evidence, novel watermark embedding process improved the robustness of watermarked object. The evaluation results of fidelity have been giving evidence of novel watermark embedding process improved the visual fidelity between original image and watermarked image. Robustness and fidelity are conflicting characteristics. It is difficult to make a balance between robustness and fidelity at the same time. Novel feature detection and embedding algorithms have provided a guarantee of both robustness and fidelity both at the same time. Capacity is another considerable characteristic of watermarking system. The evaluation chapter has done a separate evaluation using the different test cases for capacity. The evaluation results provide evidence of a good capacity provided by proposed algorithm.

6.3 Achievements

So far, the discussion can conclude, that the novel introduced watermarking system has provided extreme robustness against noise adding, filtering, rotation, and scaling attacks. Fidelity and capacity characteristics also have improved by proposed method. Thus, it can have concluded that the first objective of this study has been successfully achieved.

The watermark extraction process of the feature based watermarking system has a somewhat complex architecture as has described in methodology chapter. There are

described experiments and results of feature based extraction in Chapter 4. The watermark extraction process requires only two inputs. The first one is a watermarked image. The second input is metadata matrix. Extraction method introduced in this thesis, never required an original image. This is a main advantage of extraction algorithm, introduced in this thesis. That means, the second objective of this study has been very successfully achieved by novel introduced watermark extraction algorithm.

During this research work, a prototype application and an experimental environment have been developed. It has carried out the huge number of experiments. There have been evaluated all experiments in the evaluation process and have been done a critical review of evaluation results. That means third and final objective of this study has been successfully achieved.

It can have concluded that, this study has achieved all of the objectives by following the scientific method of research. The final conclusion is that, findings of this study have successfully solved the addressing research problem and hypothesis has been established.

6.4 Future Work

It was concluded by giving evidence, hypothesis has established. This is the best time to suggest some future works for other researchers. One possible future development is research with different JPEG compression parameters to improve the fidelity according to human vision. Another possible improvement is applied steganography techniques and the improvements of the capacity.

6.5 Summary

This chapter has given an overview of the total solution and in depth discussion about achievements of the study. This thesis has linked all chapters together and discussed methodology, experiments and evaluation done in the research process. Finally, this study has shown evidence, given solution to successfully solve the research gap and hypothesis has established.

References

[1] Charles Way Hun Fung, Antˆonio Gortan, and Walter Godoy Junior, “A Review Study on Image Digital Watermarking,” in *The Tenth International Conference on Networks*, 2011, pp. 24–28.

[2] M. L. Miller, I. J. Cox, J.-P. M. Linnartz, and T. Kalker, “A review of watermarking principles and practices,” *Digit. Signal Process. Multimed. Syst.*, pp. 461–485, 1999.

[3] H. Tao, L. Chongmin, J. M. Zain, and A. N. Abdalla, “Robust image watermarking theories and techniques: A review,” *J. Appl. Res. Technol.*, vol. 12, no. 1, pp. 122–138, 2014.

[4] Joachim von zur Gathen and El-Gayyar, “Watermarking Techniques Spatial Domain Digital Rights Seminar copyright.”

[5] C. Nafornita, A. Isar, and M. Borda, “Improved Pixel-Wise Masking for Image Watermarking,” in *Multimedia Content Representation, Classification and Security*, vol. 4105, B. Gunsel, A. K. Jain, A. M. Tekalp, and B. Sankur, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 90–97.

[6] C. Nafornita, A. Isar, and M. Borda, “Pixel-wise masking for watermarking using local standard deviation and wavelet compression,” *Sci. Bull. Politeh. Univ Timisoara Trans Electron. Telecommun.*, vol. 51, no. 65, pp. 146–151, 2006.

[7] H.-J. Wang, P.-C. Su, and C.-C. J. Kuo, “Wavelet-based digital image watermarking,” *Opt. Express*, vol. 3, no. 12, pp. 491–496, 1998.

[8] Mei Jiansheng, Li Sukang, and Tan Xiaomei, “A Digital Watermarking Algorithm Based On DCT and DWT,” *Int. Symp. Web Inf. Syst. Appl.*, pp. 22–24, May 2009.

[9] K. Loukhaoukha, “Security of ownership watermarking of digital images based on singular value decomposition,” *J. Electron. Imaging*, vol. 19, no. 1, p. 013007, Jan. 2010.

[10] K. Loukhaoukha, J.-Y. Chouinard, and M. H. Taieb, “Multi-Objective Genetic Algorithm Optimization for Image Watermarking Based on Singular Value Decomposition and Lifting Wavelet Transform,” in *Image and Signal Processing*, vol. 6134, A. Elmoataz, O. Lezoray, F. Nouboud, D. Mammass, and J. Meunier, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 394–403.

[11] K. Loukhaoukha and J.-Y. Chouinard, “Hybrid watermarking algorithm based on SVD and lifting wavelet transform for ownership verification,” 2009, pp. 177–182.

[12] D.-C. Wu and W.-H. Tsai, “A steganographic method for images by pixel-value differencing,” *Pattern Recognit. Lett.*, vol. 24, no. 9–10, pp. 1613–1626, Jun. 2003.

[13] K.-C. Chang, C.-P. Chang, P. S. Huang, and T.-M. Tu, “A novel image steganographic method using tri-way pixel-value differencing,” *J. Multimed.*, vol. 3, no. 2, pp. 37–44, 2008.

[14] X. Wu, J. Hu, Z. Gu, and J. Huang, “A secure semi-fragile watermarking for image authentication based on integer wavelet transform with parameters,” in *Proceedings of the 2005 Australasian workshop on Grid computing and e-research-Volume 44*, 2005, pp. 75–80.

[15] P. Campisi, D. Kundur, and A. Neri, “Robust Digital Watermarking in the Ridgelet Domain,” *IEEE Signal Process. Lett.*, vol. 11, no. 10, pp. 826–830, Oct. 2004.

[16] K. Manashee and T. Themrichon, “A Comparative Study of Steganography Algorithms of Spatial and Transform Domain - ncit175194.pdf,” *Int. J. Comput. Appl.*, 2015.

[17] M. Kim, D. Li, and S. Hong, “A Robust Digital Watermarking Technique for Image Contents based on DWT-DFRNT Multiple Transform Method,” *Int. J. Multimed. Ubiquitous Eng.*, vol. 9, no. 1, pp. 369–378, Jan. 2014.

[18] Z. Yuefeng and L. Li, “DIGITAL IMAGE WATERMARKING ALGORITHMS BASED ON DUAL TRANSFORM DOMAIN AND SELF-RECOVERY,” *Int. J. Smart Sens. Intell. Syst.*, vol. 8, no. 1, 2015.

[19] Z. Dawei, C. Guanrong, and L. Wenbo, “A chaos-based robust wavelet-domain watermarking algorithm,” *Chaos Solitons Fractals*, vol. 22, no. 1, pp. 47–54, Oct. 2004.

[20] N. Nikolaidis and I. Pitas, “Robust image watermarking in the spatial domain,” *Signal Process.*, vol. 66, no. 3, pp. 385–403, 1998.

[21] F. Sebé, J. Domingo-Ferrer, and J. Herrera, “Spatial-domain image watermarking robust against compression, filtering, cropping, and scaling,” in *Information Security*, Springer, 2000, pp. 44–53.

[22] E. Praun, H. Hoppe, and A. Finkelstein, “Robust mesh watermarking,” in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, 1999, pp. 49–56.

[23] A. Bamatraf, R. Ibrahim, and M. N. B. M. Salleh, “Digital watermarking algorithm using LSB,” 2010, pp. 155–159.

[24] Farid Ahmed and Ira S. Moskowitz, “Correlation-based watermarking method for image authentication applications,” *Opt. Eng.*, vol. 43, no. 8, Feb. 2004.

[25] R. L. De Queiroz, “Processing JPEG-compressed images and documents,” *Image Process. IEEE Trans. On*, vol. 7, no. 12, pp. 1661–1672, 1998.

[26] G. K. Wallace, “The JPEG still picture compression standard,” *IEEE Trans. Consum. Electron.*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[27] A. N. Skodras, C. A. Christopoulos, and T. Ebrahimi, “JPEG2000: The upcoming still image compression standard,” *Pattern Recognit. Lett.*, vol. 22, no. 12, pp. 1337–1345, 2001.

[28] H. Wood, “Invisible Digital Watermarking the Spatial and DCT Domains for Color Images,” *Adams State Coll. Alamosa Colo.*

[29] X. Qi, “An Efficient Wavelet-based Watermarking Algorithm,” in *proceedings of Hawaii International Conference on Computer Sciences*, 2002, pp. 383–388.

[30] S. Zhang and K. Yoshino, “DWT-Based Watermarking Using QR Code,” 2008.

[31] C. Nafornita, A. Isar, and M. Borda, “Image Watermarking Based on the Discrete Wavelet Transform Statistical Characteristics,” 2005, pp. 943–946.

[32] C. Nafornita, A. Isar, and M. Kovaci, “Increasing watermarking robustness using turbo codes,” 2009, pp. 113–118.

[33] J. Ayubi, S. Mohanna, F. Mohanna, and M. Rezaei, “A chaos based blind digital image watermarking in the wavelet transform domain,” *Int. J. Comput. Sci. Issues*, vol. 8, no. 4, 2011.

[34] M. N. Do and M. Vetterli, “The finite ridgelet transform for image representation,” *IEEE Trans. Image Process.*, vol. 12, no. 1, pp. 16–28, 2003.

[35] C. Harris and M. Stephens, “A Combined Corner and Edge Detector,” 1988, p. 23.1-23.6.

[36] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[37] F. Jin and D. Feng, “Image Registration Algorithm Using Mexican Hat Function-Based Operator and Grouped Feature Matching Strategy,” *PLoS ONE*, vol. 9, no. 4, p. e95576, Apr. 2014.

[38] D. K. Park, Y. S. Jeon, and C. S. Won, “Efficient use of local edge histogram descriptor,” in *Proceedings of the 2000 ACM workshops on Multimedia*, 2000, pp. 51–54.

[39] C. S. Won, D. K. Park, and S.-J. Park, “Efficient use of MPEG-7 edge histogram descriptor,” *Etri J.*, vol. 24, no. 1, pp. 23–30, 2002.

[40] J. Fridrich and M. Goljan, “Digital image steganography using stochastic modulation,” in *Electronic Imaging 2003*, 2003, pp. 191–202.

[41] B. Dirk and A. RWTH, “stochastic modulation.”

[42] J. J. Harmsen, K. D. Bowers, and W. A. Pearlman, “Fast additive noise steganalysis,” in *Electronic Imaging 2004*, 2004, pp. 489–495.

[43] Alan V. Oppenheim, Ronald W. Schafer, and John R. Buck, *Discrete time Signal Processing*, 2nd Edition.

[44] V. Senthooran and L. Ranathunga, “DCT coefficient dependent quantization table modification steganographic algorithm,” in *Networks & Soft Computing (ICNSC), 2014 First International Conference on*, 2014, pp. 432–436.

[45] A. Zigomitros and C. Patsakis, “Cross format embedding of metadata in images using QR codes,” in *Intelligent Interactive Multimedia Systems and Services*, Springer, 2011, pp. 113–121.

[46] H.-Y. Lee, I. K. Kang, H.-K. Lee, and Y.-H. Suh, “Evaluation of feature extraction techniques for robust watermarking,” in *International Workshop on Digital Watermarking*, 2005, pp. 418–431.

[47] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.

[48] Z. Wang, E. P. Simoncelli, and A. C. Bovik, “Multiscale structural similarity for image quality assessment,” in *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, 2003, vol. 2, pp. 1398–1402.

[49] C. Li and A. C. Bovik, “Three-component weighted structural similarity index,” in *IS&T/SPIE Electronic Imaging*, 2009, p. 72420Q–72420Q.

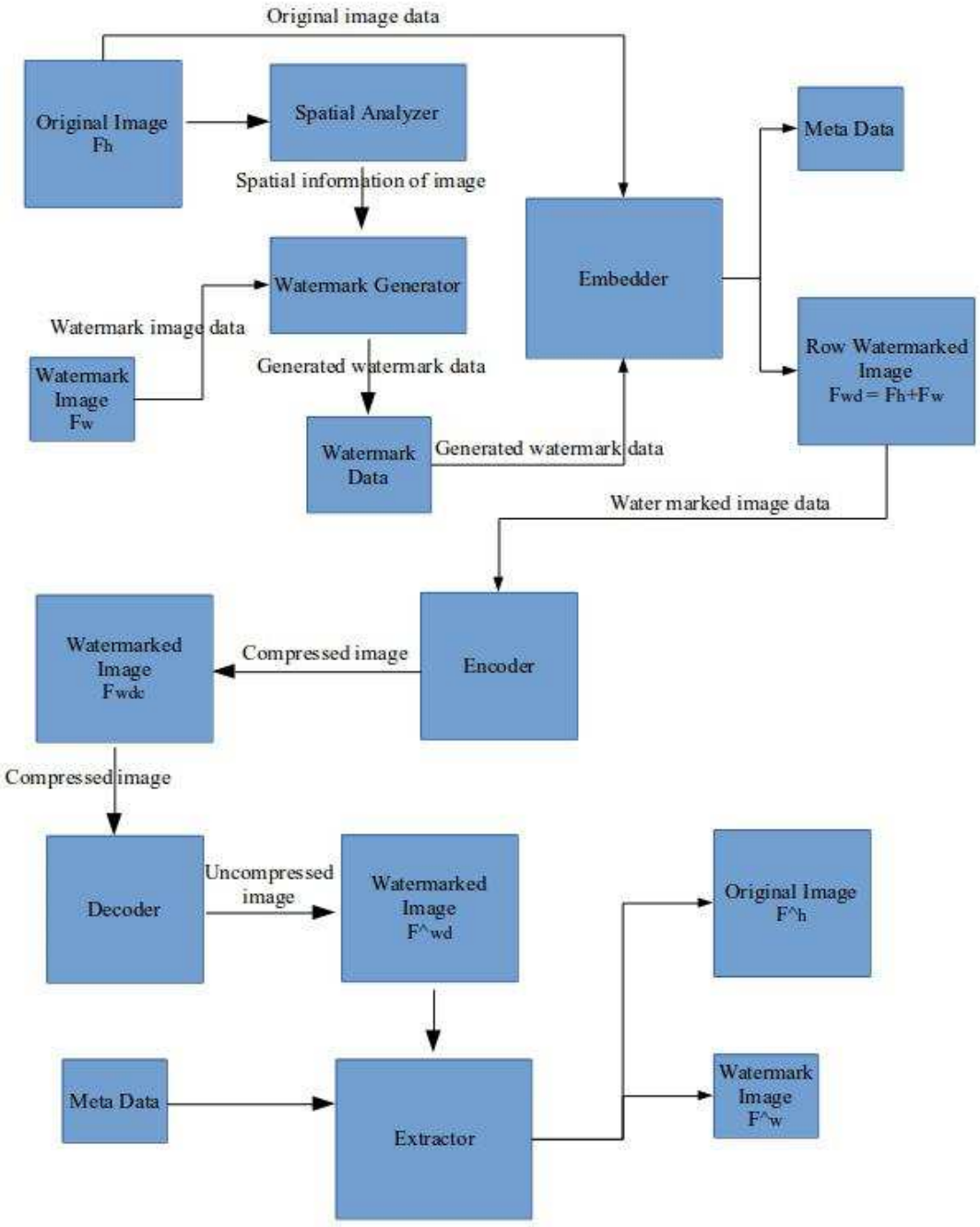
[50] Lin Zhang, Lei Zhang, Xuanqin Mou, and Daivd Zhang, “FSIM: A Feature Similarity Index for Image Quality Assessment.”

[51] T. Shohdohji, Y. Hoshino, and N. Kutsuwada, “Optimization of quantization table based on visual characteristics in DCT image coding,” *Comput. Math. Appl.*, vol. 37, no. 11–12, pp. 225–232, Jun. 1999.

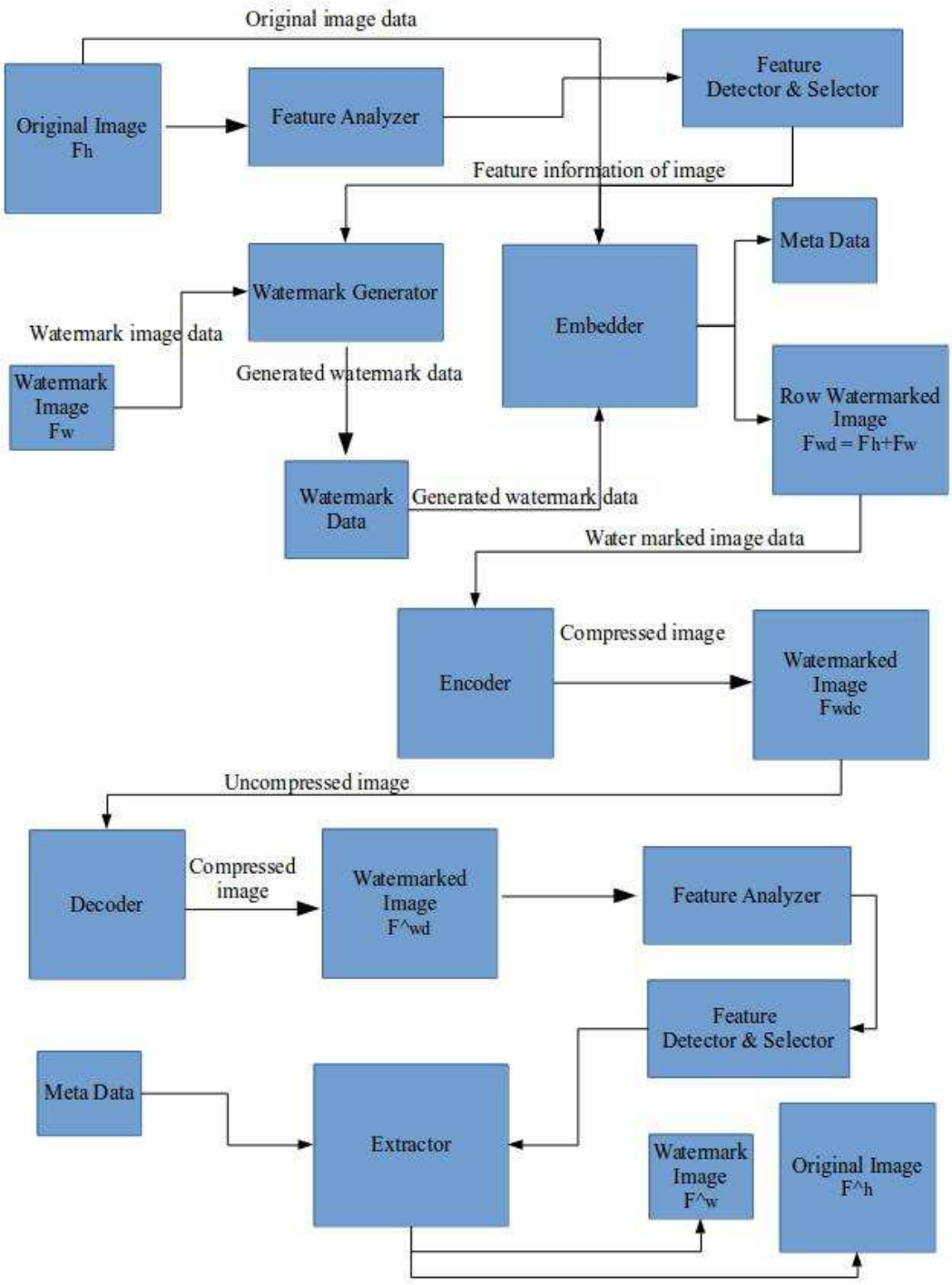
[52] Q. Li, C. Yuan, and Y. Z. Zhong, “Adaptive DWT-SVD Domain Image Watermarking Using Human Visual Model,” in *the 9th International Conference on*

Advanced Communication Technology, 2007, vol. 3, pp. 1947–1951.

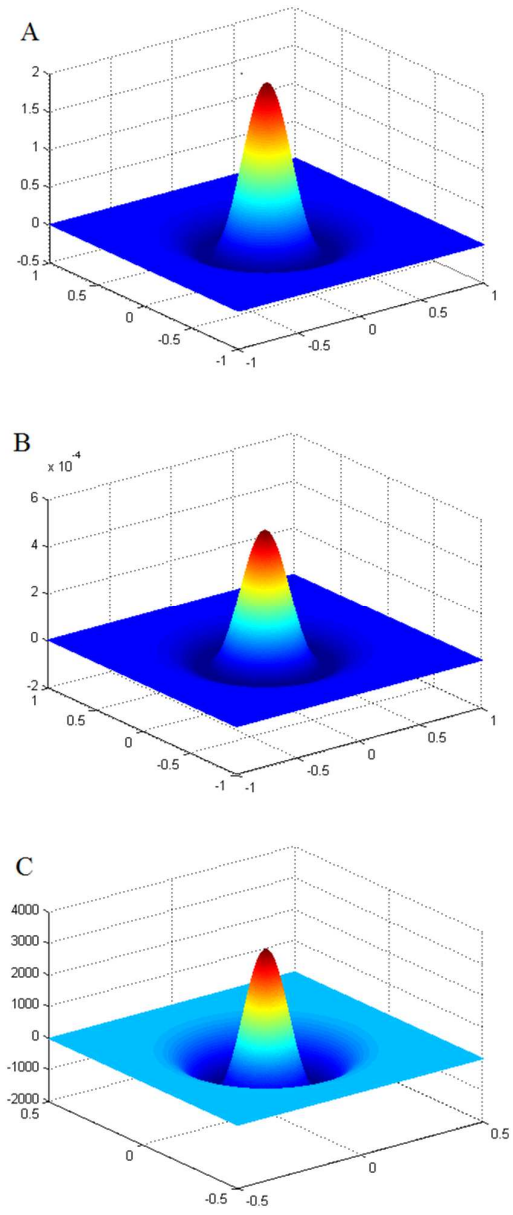
Appendix A (Methodology)



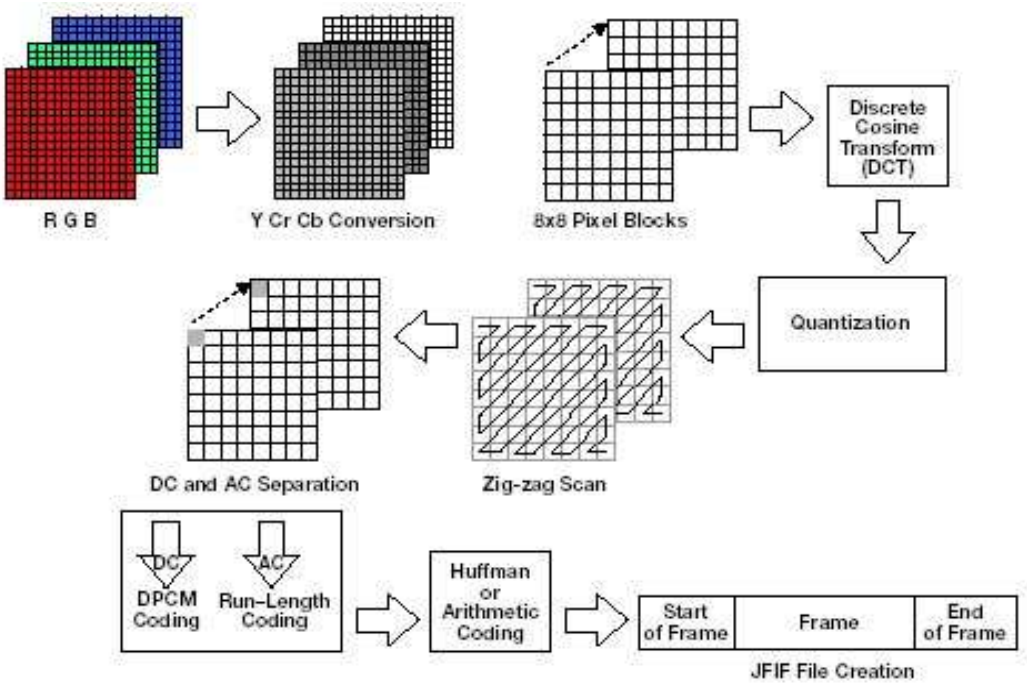
Appendix A, Figure 1: High level design diagram of pixel base watermarking system



Appendix A, Figure 2: High level design diagram of feature base watermarking system



Appendix A, Figure 3: Mexican hat operator with different sigma values. (Source: http://journals.plos.org)



Appendix A, Figure 3: Encoder module. *(Source: http://www.eetimes.com/)*

Appendix B (Experimental Design)

Dataset Used in Research

Name: lena1.bmp

Format: bit map

Dimension: 640x640 Px

Size: 1.2Mb

Name: lena2.jpg

Format: jpg

Dimension: 512x512 Px

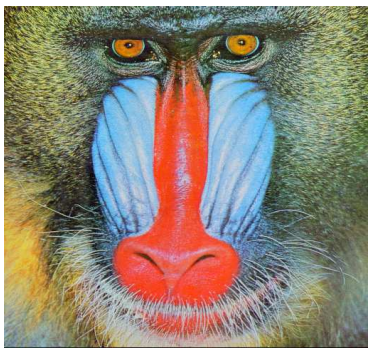
Size: 94.4 Kb

Name: lena3.png

Format: png

Dimension: 512x512 Px

Size: 473.8 Kb



Name: monky.png

Format: png

Dimension: 512x512 Px

Size: 626.9 Kb

Name: veg.jpg

Format: jpg

Dimension: 512x512 Px

Size: 48.5 Kb

Experimental results of pixel based watermark

Experiment 1: Embed using random insertion

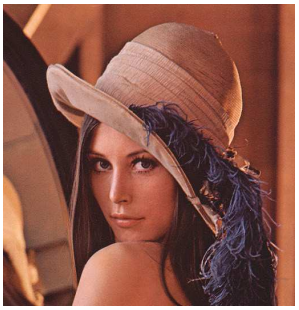
Inputs		Outputs	
Base method	: Pixel base	Watermarked image	: result_lena.jpg
Technique	: Random insertion	Metadata file	: result_lena.csv
Original image	: lena.jpg		
Watermark	: Watermark 4		

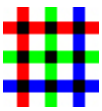
Compression parameters:

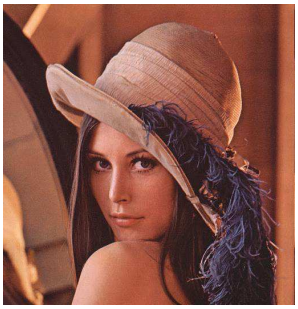
Block size: 8x8

Quality: 70%

JSAMPROW row pointer size: 1







Original image	Watermark image	Watermarked image
Dimension: 512x512	Dimension: 48x48	Dimension: 512x512
Size: 94.4 Kb	Size: 22.8 Kb	Size: 37.9 Kb

Experiment 2: Embed using less sensitive points of human vision

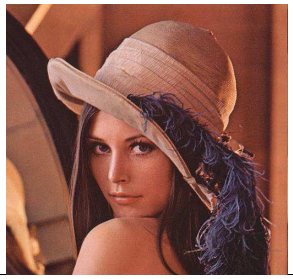
Inputs		Outputs	
Base method	: Pixel base	Watermarked image	: result_lena.jpg
Technique	: less sensitive points of human vision	Metadata file	: result_lena.csv
Original image	: lena.jpg		
Watermark	: Watermark 4		

Compression parameters:

Block size: 8x8

Quality: 70%

JSAMPROW row pointer size: 1



Original image	Watermark image	Watermarked image
Dimension: 512x512	Dimension: 48x48	Dimension: 512x512
Size: 94.4 Kb	Size: 22.8 Kb	Size: 37.9 Kb

Experiment 3: Embed using LSB of macro-block

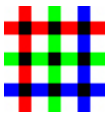
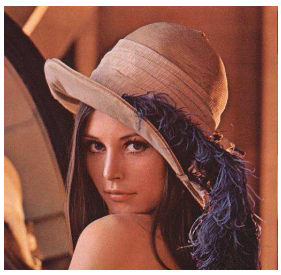
Inputs		Outputs	
Base method	: Pixel base	Watermarked image	: result_lena.jpg
Technique	: LSB of macro-block	Metadata file	: result_lena.csv
Original image	: lena.jpg		
Watermark	: Watermark 4		

Compression parameters:

Block size: 8x8

Quality: 70%

JSAMPROW row pointer size: 1

Original image Dimension: 512x512 Size: 37.9 Kb	Watermark image Dimension: 48x48 Size: 22.8 Kb	Watermarked image Dimension: 512x512 Size: 45.4 Kb

Experimental results of pixel based watermark extraction methods

Experiment 4: Extraction using common algorithm

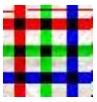
Inputs	Outputs
Base method : Pixel base Technique : common Original image : result_lena.jpg Metadata file : result_lena.csv	Extracted watermark: wtm.jpg

Compression parameters:

Block size: 8x8

Quality: 70%

JSAMPROW row pointer size: 1

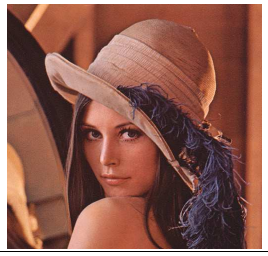
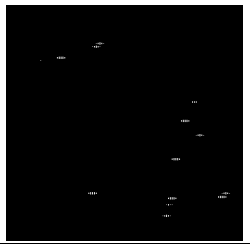
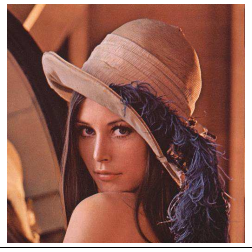



Watermarked image Dimension: 512x512 Size: 45.4 Kb	Extracted watermark Dimension: 48x48 Size: 1.9 Kb

Experimental results of feature based watermark extraction methods

Experiment 5: Embed using Harris corner detector

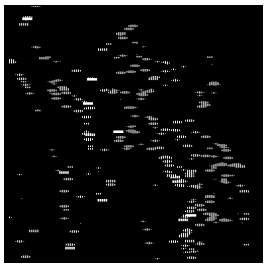
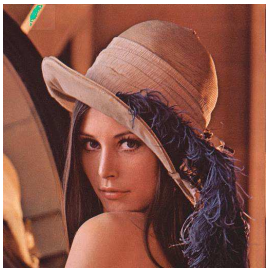
Inputs	Base method : Feature base Possession : Around single corner Original image : lena.jpg Watermark : Watermark 4
Outputs	Watermarked image : result_lena.jpg Metadata file : result_lena.csv
Settings	Feature detector: Harris operator Smoothing: Gaussian filter Kernel: 5x5 Sigma: 1 Threshold: 120
Compression parameters	Block size: 8x8 Quality: 70% JSAMPROW row pointer size: 1

		
Original image Dimension: 512x512 Size: 37.9 Kb	Binary corner image Dimension: 512x512 Size: 16.2 Kb	Watermarked image Dimension: 512x512 Size: 47.4 Kb

Experiment 6: Embed using novel corner detector

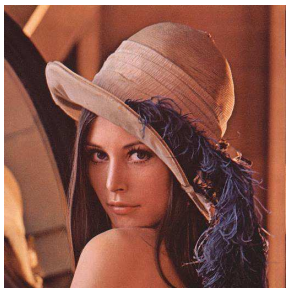
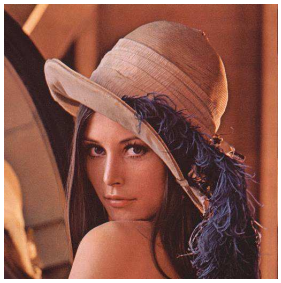
Inputs	Base method : Feature base Position : Around single corner Original image : lena.jpg Watermark : Watermark 4
Outputs	Watermarked image : result_lena.jpg Metadata file : result_lena.csv
Settings	Feature detector: Novel operator Smoothing: LoG

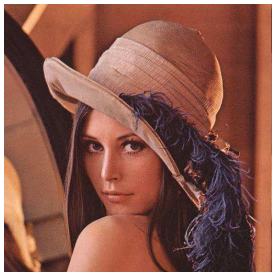
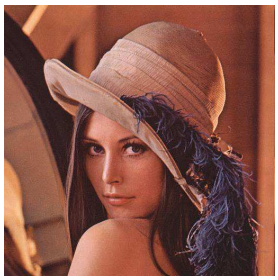
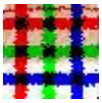
	Kernel: 5x5 Sigma: 1 Threshold: 120
Compression parameters	Block size: 8x8 Quality: 70% JSAMPROW row pointer size: 1

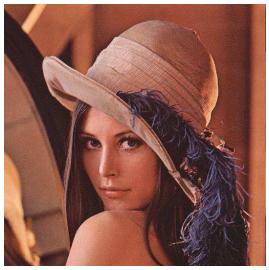
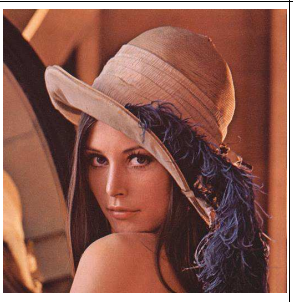
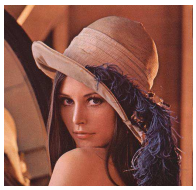
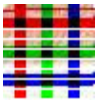
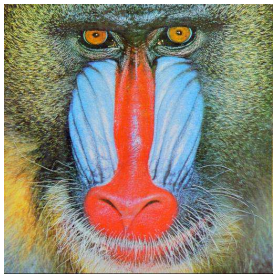
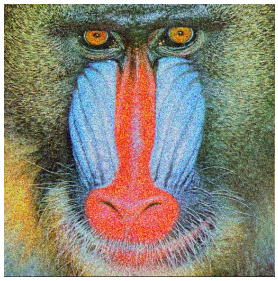
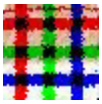
		
Original image Dimension: 512x512 Size: 37.9 Kb	Binary corner image Dimension: 512x512 Size: 18.3 Kb	Watermarked image Dimension: 512x512 Size: 46.2 Kb

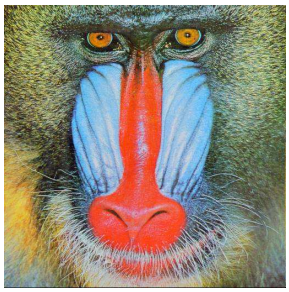
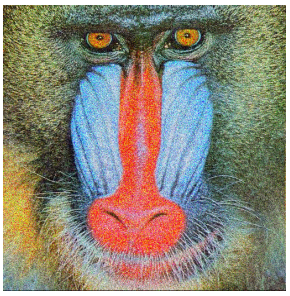
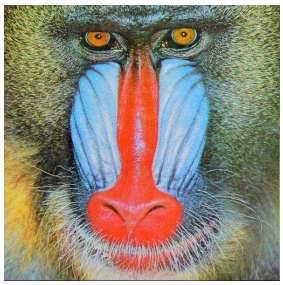
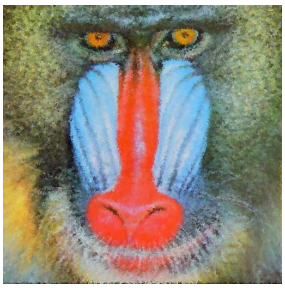
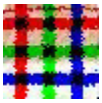
Appendix C (Evaluation)

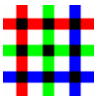
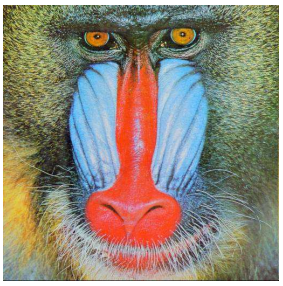
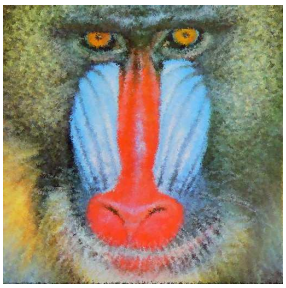
Evaluate of the Robustness

Settings & Attack	Watermark	Watermarked Image	Watermarked Image After Attack
Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner **HSV noise** [Holdness: 3] [Hue:72] [Saturation:146] [Value: 94]		 	 
Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner **HSV noise** [Holdness: 3] [Hue:72] [Saturation:146] [Value: 94]		 	 

Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1.5 Threshold: 125 Position: single corner **Random Pik.** [Random seeds: 1452988117] [Randomization: 42] [Repeat: 5]		 	 
Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1.5 Threshold: 125 Position: single corner **Random Pik.** [Random seeds: 1452988117] [Randomization: 42] [Repeat: 5]		 	 

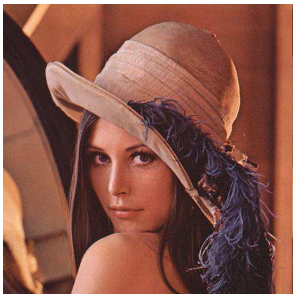
Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1.5 Threshold: 125 Position: single corner **Rotation in 45 degree**		 	 
Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1.5 Threshold: 125 Position: single corner **Scaling** watermarked image: 512x512 scaled to: 256x256		 	 
Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner **HSV noise** [Holdness: 3] [Hue:72] [Saturation:146] [Value: 94]		 	 

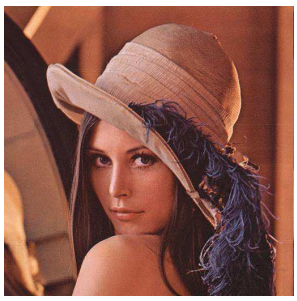
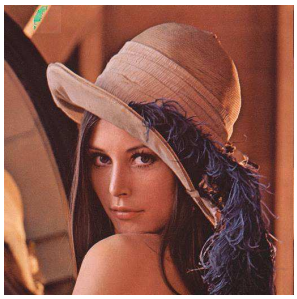
Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner **HSV noise** [Holdness: 3] [Hue:72] [Saturation:146] [Value: 94]		 	 
Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner **Random Pik.** [Random seeds: 1452988117] [Randomization: 42] [Repeat: 5]		 	 

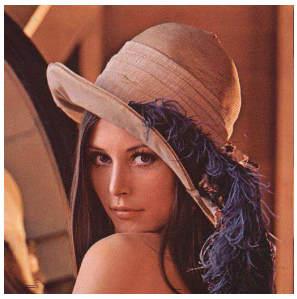
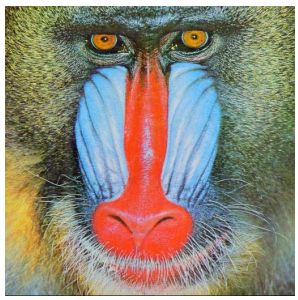
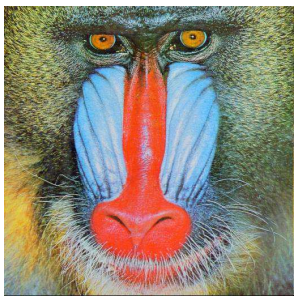
Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner **Random Pik.** [Random seeds: 1452988117] [Randomization: 42] [Repeat: 5]		 	 

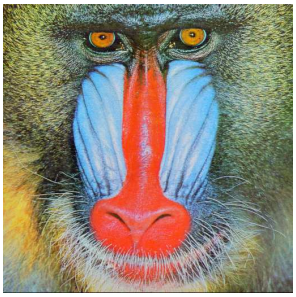
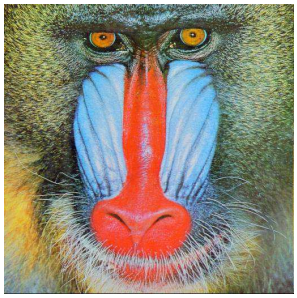
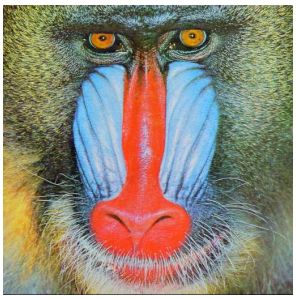
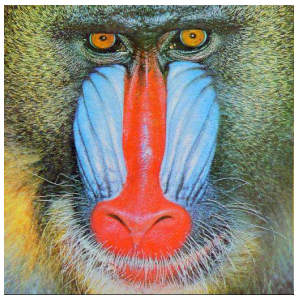
Appendix C, Table 1: Summary of evaluation results for the robustness

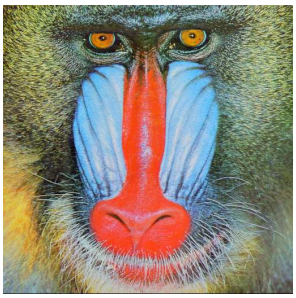
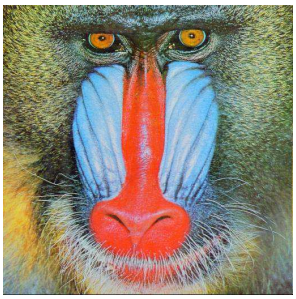
Evaluate of the Fidelity

Image Detail & Settings	Evaluation Results	Original Image	Watermarked Image
Image: lena.jpg (512x512, 94.4 Kb) **Settings** Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner	**Evaluation Method 1: MSE** Resulting Value: 102.03 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 39.28 Conclusion: Good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel[0]: 0.69 Chanel[1]: 0.75 Chanel[2]: 0.74 Mean: 0.72 Conclusion: Good		

	Fidelity		
Image: lena.jpg (512x512, 94.4 Kb) **Settings** Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	**Evaluation Method 1: MSE** Resulting Value: 113.03 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 41.28 Conclusion: Good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel[0]: 0.69 Chanel[1]: 0.75 Chanel[2]: 0.74 Mean: 0.72 Conclusion: Good Fidelity		
Image: lena.jpg (512x512, 94.4 Kb) **Settings** Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner	**Evaluation Method 1: MSE** Resulting Value: 76.03 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 44.36 Conclusion: Very good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel[0]: 0.79 Chanel[1]: 0.85		

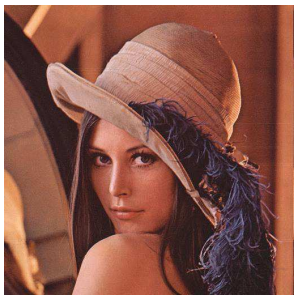
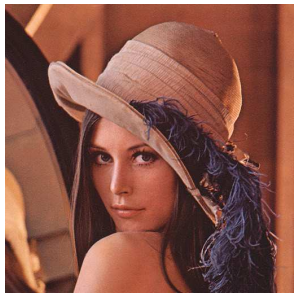
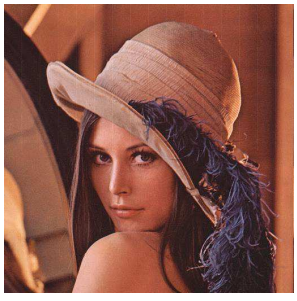
	Chanel[2]: 0.84 Mean: 0.82 Conclusion: Very good Fidelity		
Image: lena.jpg (512x512, 94.4 Kb) **Settings** Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	**Evaluation Method 1: MSE** Resulting Value: 54.44 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 45.51 Conclusion: Very good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel[0]: 0.79 Chanel[1]: 0.85 Chanel[2]: 0.84 Mean: 0.82 Conclusion: Very good Fidelity		
Image: monky.png (512x512, *626.9 Kb*) **Settings** Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: single corner	**Evaluation Method 1: MSE** Resulting Value: 202.03 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 38.80 Conclusion: Good fidelity **Evaluation Method 3: SSIM**		

	Resulting Value: Chanel[0]: 0.69 Chanel[1]: 0.75 Chanel[2]: 0.74 Mean: 0.72 Conclusion: Good Fidelity		
Image: monky.png (512x512, *626.9* Kb) **Settings** Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	**Evaluation Method 1: MSE** Resulting Value: 203.03 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 38.28 Conclusion: Good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel[0]: 0.69 Chanel[1]: 0.75 Chanel[2]: 0.74 Mean: 0.72 Conclusion: Good Fidelity		
Image: monky.png (512x512, *626.9* Kb) **Settings** Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125	**Evaluation Method 1: MSE** Resulting Value: 146.03 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 44.06 Conclusion: Very good fidelity		

Position: single corner	**Evaluation Method 3: SSIM** Resulting Value: Chanel[0]: 0.79 Chanel[1]: 0.85 Chanel[2]: 0.84 Mean: 0.82 Conclusion: Very good Fidelity		
Image: monky.png (512x512, *626.9* Kb) **Settings** Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	**Evaluation Method 1: MSE** Resulting Value: 54.44 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 45.01 Conclusion: Very good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel[0]: 0.80 Chanel[1]: 0.85 Chanel[2]: 0.85 Mean: 0.84 Conclusion: Very good Fidelity		

Appendix C, Table 2: Summary of evaluation results for the fidelity

Evaluate of the Capacity

Image Detail & Settings	Evaluation Results (Statistically)	Evaluation Results (Experimentally) *Original Image*	*Evaluation Results (Experimentally) Watermarked Image*
Image: lena.jpg (512x512, 94.4 Kb) **Watermark object:** Name: watermark 4.jpg Dimension: 48x48 px, Size: 22.8 Kb **Settings** Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corner	**Evaluation Method 1: MSE** Resulting Value: 103.03 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 41.28 Conclusion: Good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel[0]: 0.69 Chanel[1]: 0.75 Chanel[2]: 0.74 Mean: 0.72 Conclusion: Good Fidelity		
Image: lena.jpg (512x512, 94.4 Kb) **Watermark object:** Name: watermark 4.jpg Dimension: 64x64 px, Size: 27.6 Kb	**Evaluation Method 1: MSE** Resulting Value: 183.04 Conclusion: Fair fidelity **Evaluation Method 2: PSNR** Resulting Value: 37.44 Conclusion: Fair fidelity **Evaluation Method 3: SSIM**		

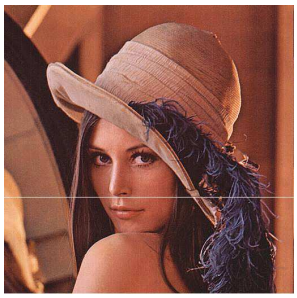
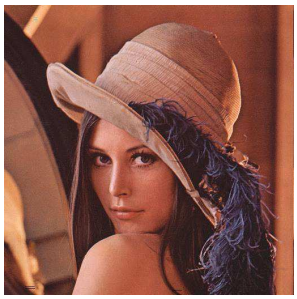
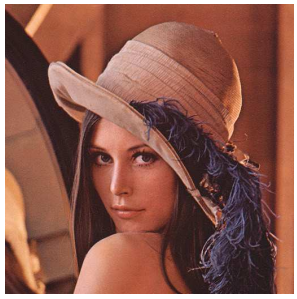
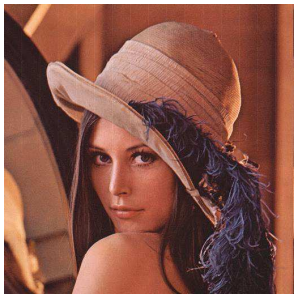
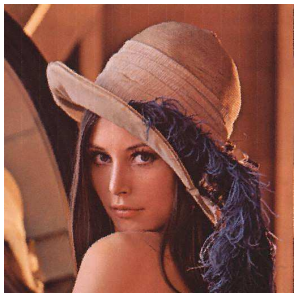
Settings Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	Resulting Value: Chanel[0]: 0.69 Chanel[1]: 0.75 Chanel[2]: 0.74 Mean: 0.72 Conclusion: Good Fidelity		
Image: lena.jpg (512x512, 94.4 Kb) **Watermark object:** Name: watermark 4.jpg Dimension: 128x128 px, Size: 29.8 Kb **Settings** Feature detector: Harris operator Smoothing: Gaussian Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	**Evaluation Method 1: MSE** Resulting Value: 344.05 Conclusion: Very bad fidelity **Evaluation Method 2: PSNR** Resulting Value: 23.82 Conclusion: Very bad fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel[0]: 0.59 Chanel[1]: 0.64 Chanel[2]: 0.66 Mean: 0.63 Conclusion: Bad Fidelity		

Image: lena.jpg (512x512, 94.4 Kb) **Watermark object:** Name: watermark 4.jpg Dimension: 48x48 px, Size: 22.8 Kb **Settings** Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	**Evaluation Method 1: MSE** Resulting Value: 54.44 Conclusion: Very good fidelity **Evaluation Method 2: PSNR** Resulting Value: 45.51 Conclusion: Very good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel[0]: 0.79 Chanel[1]: 0.85 Chanel[2]: 0.84 Mean: 0.82 Conclusion: Very good Fidelity		
Image: lena.jpg (512x512, 94.4 Kb) **Watermark object:** Name: watermark 4.jpg Dimension: 64x64 px, Size: 27.6 Kb **Settings** Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1	**Evaluation Method 1: MSE** Resulting Value: 143.04 Conclusion: Good fidelity **Evaluation Method 2: PSNR** Resulting Value: 40.04 Conclusion: Good fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel[0]: 0.69 Chanel[1]: 0.75 Chanel[2]: 0.70 Mean: 0.71 Conclusion: Good Fidelity		

Threshold: 125 Position: multiple corners			
Image: lena.jpg (512x512, 94.4 Kb) **Watermark object:** Name: watermark 4.jpg Dimension: 128x128 px, Size: 27.6 Kb **Settings** Feature detector: Novel operator Smoothing: LoG Kernel: 5x5 Sigma: 1 Threshold: 125 Position: multiple corners	**Evaluation Method 1: MSE** Resulting Value: 163.04 Conclusion: Bad fidelity **Evaluation Method 2: PSNR** Resulting Value: 40.04 Conclusion: Fair fidelity **Evaluation Method 3: SSIM** Resulting Value: Chanel[0]: 0.60 Chanel[1]: 0.65 Chanel[2]: 0.66 Mean: 0.64 Conclusion: Fair Fidelity		

Appendix C, Table 3: Summary of evaluation results for the capacity

Appendix D (Prototype System)

Structure of Prototype Application

/ ---------- root
/CMake Files ---------- build files

/GUI ---------- GUI of application
/GUI/images ---------- application images
/GUI/source ---------- source-code of GUI application
main.cpp

/GUI/source/class ---------- C++ class files of GUI application
evaluate.cpp
mainwindow.cpp
featurewindow.cpp
pixelwindow.cpp

/GUI/source/headers ---------- C++ header files of GUI application
mainwindow.h

/GUI/source/includes ---------- custom header files
headers.h

/GUI/gui.pro ---------- configuration file

/images ---------- input image
/result ---------- output
/source ---------- source code of watermarking application
main.cpp

/source/compress ---------- encoder algorithms

render_jpeg.cpp

/source/embed ---------- embedding algorithms

embed_watermark_corners_harris.cpp

embed_watermark_corners_lochandaka.cpp

embed_watermark_corners.cpp

embed_watermark_pixel_full_fixed_inblock_position.cpp

embed_watermark_pixel_full_fixed_position.cpp

embed_watermark_pixel_full_vari_inblock_position.cpp

embed_watermark_pixel.cpp

/source/evaluate ---------- evaluation methods

mse.cpp

psnr.cpp

ssim.cpp

/source/extract ---------- extraction algorithms

extract_watermark_corners_harris.cpp

extract_watermark_corners_lochandaka.cpp

extract_watermark_corners.cpp

extract_watermark_pixel_full_common_position.cpp

extract_watermark_pixel_full_fixed_inblock_position.cpp

extract_watermark_pixel_full_fixed_position.cpp

extract_watermark_pixel_full_vari_inblock_position.cpp

extract_watermark_pixel.cpp

/source/header ---------- headers

function_decleration.h

includes.h

typedef.h

/source/operators ---------- feature detection operators

harris_operator.cpp

lochandaka_operator.cpp

/source/read ---------- read image data

read_jpeg.cpp

read_image.cpp

read_jpeg.cpp

read_pixel.cpp

/source/res ---------- mathematical functions

convert_color_space.cpp

draw_circle.cpp

drow_image.cpp

filters.cpp

get_corner_points.cpp

get_image_dimensions.cpp

intensity_diff.cpp

math.cpp

pixel_intensity_boundary_adjust.cpp

read_watermark_meta.cpp

Total Lines of code: 6212

Main Screens of Prototype Application

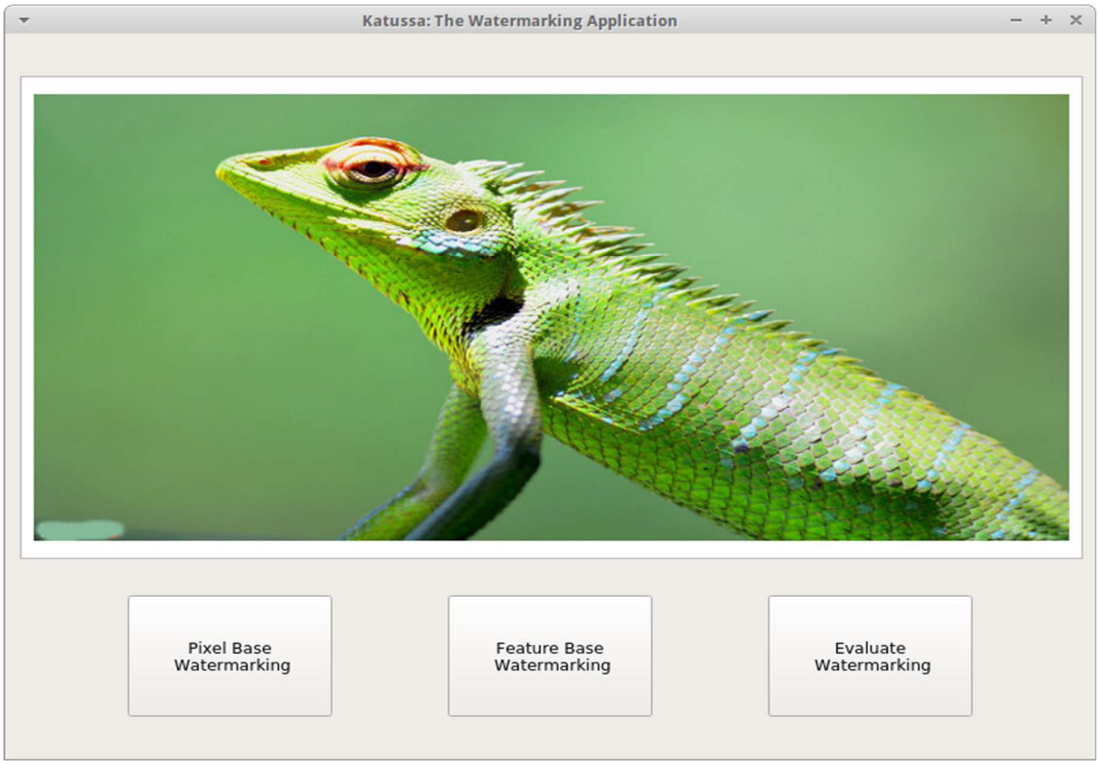
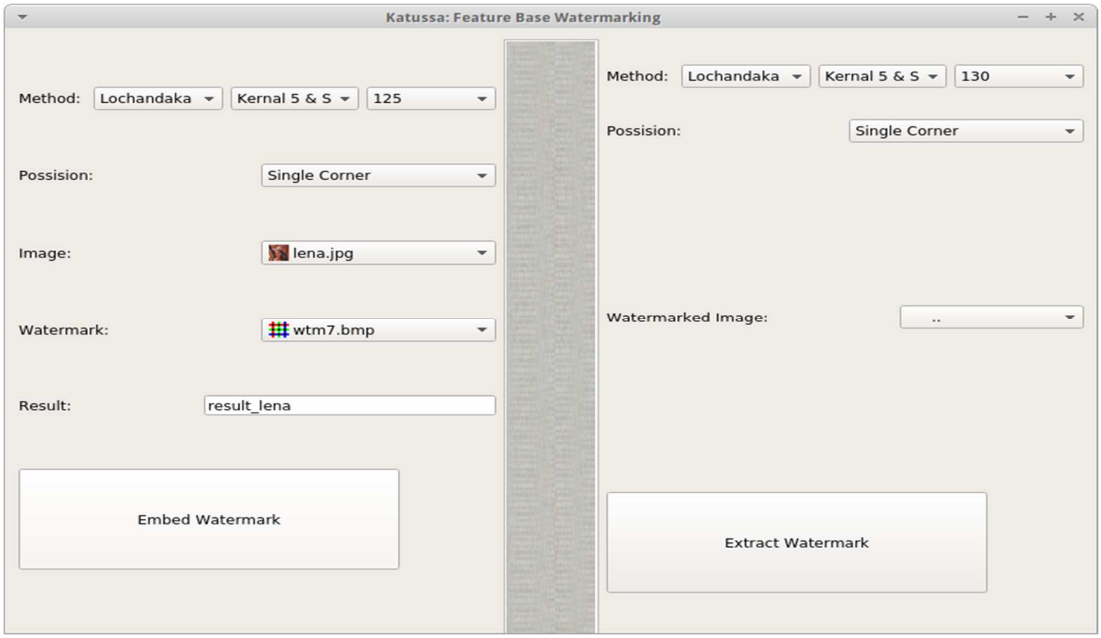


Fig. 1. Appendix D, Figure 1: Main window of prototype system



Appendix D, Figure 2: Watermark embed & extraction window

Appendix D, Figure 2: *Evaluation window*

Source code of novel watermark operators

```
/* Create namespace */
using namespace cv;
using namespace std;

DERIVATIVES katussa_sobel_operator(Mat image);
Mat katussa_gauss_operator_for_lochandaka(Mat IX, int kernal, float sigma);
Mat katussa_lochandaka_response(Mat Gsx, Mat Gsy, Mat Gsxy, float k);
Mat katussa_nonmaximal_suppression_for_lochandaka(Mat response, int threshold);

/**
 * This function implement Harris operator
 * Parameters are image, threshold
 * Return type Mat
 *
 */
Mat katussa_lochandaka_operator(Mat image, float k, int kernal, float sigma, int threshold)
{

        Mat res_img(image.rows, image.cols, CV_8UC1, Scalar(0));
        Mat res_img_maxsupp(image.rows, image.cols, CV_8UC1, Scalar(0));
        Mat Gsx(image.rows, image.cols, CV_32F);
        Mat Gsy(image.rows, image.cols, CV_32F);
        Mat Gsxy(image.rows, image.cols, CV_32F);
        Mat response(image.rows, image.cols, CV_32F);
        DERIVATIVES derivatives;

        // #### (1) Compute X and Y derivatives and compute products of derivatives
```

```
	derivatives = katussa_sobel_operator(image);

	// #### (2) Appy Gaussian operaator to Ix, Iy, Ixy
	Gsx = katussa_gauss_operator_for_lochandaka(derivatives.Ix, kernal,  sigma);
	Gsy = katussa_gauss_operator_for_lochandaka(derivatives.Iy, kernal, sigma);
	Gsxy = katussa_gauss_operator_for_lochandaka(derivatives.Ixy, kernal, sigma);

	// #### (3) Creare matrix for each pixel(x,y) and compute response
	response = katussa_lochandaka_response(Gsx, Gsy, Gsxy, k);

	// #### (4) Non Maximum suppression of response
	res_img_maxsupp = katussa_nonmaximal_suppression_for_lochandaka(response, threshold);

	res_img = res_img_maxsupp;
	return res_img;
}

/**
 * This function implement sobel operator
 * Parameters is Mat image
 * Return type DERIVATIVES
 *
 */
DERIVATIVES katussa_sobel_operator(Mat image)
{

	float derivative_Sx;
	float derivative_Sy;
	float DSx_2 = 0, DSy_2 = 0, DSxy = 0;

	Mat Ix(image.rows, image.cols, CV_32F);
	//Mat Ix(image.rows, image.cols, CV_8UC1, Scalar(0));
	Mat Iy(image.rows, image.cols, CV_32F);
	Mat Ixy(image.rows, image.cols, CV_32F);
	DERIVATIVES sobel_derivatives;

	// Sobel matrix in X direction
	int Sx[3][3] = {
		{ -1, 0, 1 },
		{ -2, 0, 2 },
		{ -1, 0, 1 }
	};

	// Sobel matrix in Y direction
	/*int Sy[3][3] = {
		{ 1, 2, 1 },
		{ 0, 0, 0 },
		{ -1, -2, -1 }
	};*/
	int Sy[3][3] = {
		{ -1, -2, -1 },
		{ 0, 0, 0 },
		{ 1, 2, 1 }
	};

	for (int y=1; y<image.rows-1; y++)
	{
		for (int x=1; x<image.cols-1; x++)
		{
```

```
                        // Compute X derivatives
                        derivative_Sx = 0.0;
                        for (int Sx_y = 0; Sx_y <= 2; Sx_y++)
                        {
                                for (int Sx_x = 0; Sx_x <= 2; Sx_x++)
                                {
                                        derivative_Sx += image.at<uchar>(y + (Sx_y-1), x + (Sx_x-1))
* Sx[Sx_y][Sx_x];
                                }
                        }
                        derivative_Sx = abs(derivative_Sx);
                        //DSx_2 = derivative_Sx*derivative_Sx;
                        Ix.at<uchar>(y, x) = derivative_Sx;
                        //printf("%d ", Ix.at<uchar>(y, x));

                        // Compute Y derivatives
                        derivative_Sy = 0.0;
                        for (int Sy_y = 0; Sy_y <= 2; Sy_y++)
                        {
                                for (int Sy_x = 0; Sy_x <= 2; Sy_x++)
                                {
                                        derivative_Sy += image.at<uchar>(y + (Sy_y-1), x + (Sy_x-1))
* Sy[Sy_y][Sy_x];
                                }
                        }
                        //printf("%.2f ", derivative_Sy);
                        derivative_Sy = abs(derivative_Sy);
                        //DSx_2 = derivative_Sy*derivative_Sy;
                        Iy.at<uchar>(y, x) = derivative_Sy;
                        //printf("%d ", Iy.at<uchar>(y, x));

                        DSxy = derivative_Sx*derivative_Sy;
                        Ixy.at<float>(y, x) = DSxy;

                }
        }

        sobel_derivatives.Ix = Ix;
        sobel_derivatives.Iy = Iy;
        sobel_derivatives.Ixy = Ixy;

        return sobel_derivatives;
}


/**
 * This function implement gauss operator
 * Parameters is Mat sobel_derivatives
 * Return type Mat
 *
 */
Mat katussa_gauss_operator_for_lochandaka(Mat IX, int kernal, float sigma)
{

        Mat Gs_Mat(IX.rows, IX.cols, CV_32F);
        //int Gs[7][7];
        //float Gs_multiplier;
        float Gs_Ivalue;
```

```
	if(kernal==5 && sigma==1){
		int Gs[5][5] = {
			{ 1, 4, 7, 4, 1 },
			{ 4, 16, 26, 16, 4 },
			{ 7, 26, 41, 26, 7 },
			{ 4, 16, 26, 16, 4 },
			{ 1, 4, 7, 4, 1 }
		};
		float Gs_multiplier = 1/273;

		for (int y=2; y<IX.rows-2; y++)
		{
			for (int x=2; x<IX.cols-2; x++)
			{

				Gs_Ivalue = 0.0;
				for (int Gs_y = 0; Gs_y <= 4; Gs_y++)
				{
					for (int Gs_x = 0; Gs_x <= 4; Gs_x++)
					{
						Gs_Ivalue += IX.at<float>(y + (Gs_y-2), x + (Gs_x-
2)) * Gs[Gs_y][Gs_x];
						//Gs_Ivalue = IX.at<float>(y + (Gs_y-2), x + (Gs_x-
2));
						//Gs_Ivalue2 += Gs_Ivalue * Gs[Gs_y][Gs_x];
					}
				}
				Gs_Mat.at<float>(y, x) = Gs_Ivalue*Gs_multiplier;

			}
		}
	}

	if(kernal==5 && sigma==2){
		int Gs[5][5] = {
			{ 2, 7, 12, 7, 2 },
			{ 7, 31, 52, 31, 7 },
			{ 12, 52, 127, 52, 12 },
			{ 7, 31, 52, 31, 7 },
			{ 2, 7, 12, 7, 2 }
		};
		float Gs_multiplier = 1/571;

		for (int y=2; y<IX.rows-2; y++)
		{
			for (int x=2; x<IX.cols-2; x++)
			{

				Gs_Ivalue = 0.0;
				for (int Gs_y = 0; Gs_y <= 4; Gs_y++)
				{
					for (int Gs_x = 0; Gs_x <= 4; Gs_x++)
					{
						Gs_Ivalue += IX.at<float>(y + (Gs_y-2), x + (Gs_x-
2)) * Gs[Gs_y][Gs_x];
						//Gs_Ivalue = IX.at<float>(y + (Gs_y-2), x + (Gs_x-
2));
						//Gs_Ivalue2 += Gs_Ivalue * Gs[Gs_y][Gs_x];
					}
```

```
                                        }
                                        Gs_Mat.at<float>(y, x) = Gs_Ivalue*Gs_multiplier;

                                }
                        }
                }

                if(kernal==5 && sigma==1.5){
                        int Gs[5][5] = {
                                { 2, 4, 5, 4, 2 },
                                { 4, 9, 12, 9, 4 },
                                { 5, 12, 15, 12, 5 },
                                { 4, 9, 12, 9, 4 },
                                { 2, 4, 5, 4, 2 }
                        };
                        float Gs_multiplier = 1/159;

                        for (int y=2; y<IX.rows-2; y++)
                        {
                                for (int x=2; x<IX.cols-2; x++)
                                {

                                        Gs_Ivalue = 0.0;
                                        for (int Gs_y = 0; Gs_y <= 4; Gs_y++)
                                        {
                                                for (int Gs_x = 0; Gs_x <= 4; Gs_x++)
                                                {
                                                        Gs_Ivalue += IX.at<float>(y + (Gs_y-2), x + (Gs_x-
2)) * Gs[Gs_y][Gs_x];
                                                        //Gs_Ivalue = IX.at<float>(y + (Gs_y-2), x + (Gs_x-
2));
                                                        //Gs_Ivalue2 += Gs_Ivalue * Gs[Gs_y][Gs_x];
                                                }
                                        }
                                        Gs_Mat.at<float>(y, x) = Gs_Ivalue*Gs_multiplier;

                                }
                        }
                }

                if(kernal==7 && sigma==1){

                        int Gs[7][7] = {
                                { 1, 4, 7, 4, 1 },
                                { 4, 16, 26, 16, 4 },
                                { 7, 26, 41, 26, 7 },
                                { 4, 16, 26, 16, 4 },
                                { 1, 4, 7, 4, 1 }
                        };
                        float Gs_multiplier = 1/273;

                        for (int y=2; y<IX.rows-2; y++)
                        {
                                for (int x=2; x<IX.cols-2; x++)
                                {

                                        Gs_Ivalue = 0.0;
                                        for (int Gs_y = 0; Gs_y <= 4; Gs_y++)
                                        {
```

```
                                        for (int Gs_x = 0; Gs_x <= 4; Gs_x++)
                                        {
                                                Gs_Ivalue += IX.at<float>(y + (Gs_y-2), x + (Gs_x-
2)) * Gs[Gs_y][Gs_x];
                                                //Gs_Ivalue = IX.at<float>(y + (Gs_y-2), x + (Gs_x-
2));
                                                //Gs_Ivalue2 += Gs_Ivalue * Gs[Gs_y][Gs_x];
                                        }
                                }
                                Gs_Mat.at<float>(y, x) = Gs_Ivalue*Gs_multiplier;

                        }
                }
        }

        return Gs_Mat;




}

/**
 * This function implement gauss operator
 * Parameters are Mat Gsx, Mat Gsy, Mat Gsxy
 * Return type Mat
 *
 */
Mat katussa_lochandaka_response(Mat Gsx, Mat Gsy, Mat Gsxy, float k)
{

        Mat response(Gsx.rows, Gsx.cols, CV_32F);
        float a11, a12, a21, a22;
        float det;
        float trace;

        for (int y=0; y<Gsx.rows; y++)
        {
                for (int x=0; x<Gsx.cols; x++)
                {

                        a11 = Gsx.at<float>(y, x) * Gsx.at<float>(y, x);
                        a22 = Gsy.at<float>(y, x) * Gsy.at<float>(y, x);
                        a12 = Gsxy.at<float>(y, x);
                        a21 = Gsxy.at<float>(y, x);

                        det = (a11*a22) - (a12*a21);
                        trace = a11 + a22;

                        //printf("%f ", det);
                        response.at<float>(y,x) = abs( (det) - (k * (trace*trace)) ); //abs( Gsx.at<float>(y,
x)*Gsy.at<float>(y, x) );
                        //printf("%f ", response.at<float>(y,x));

                }
        }

        return response;
}
```

```
Mat katussa_nonmaximal_suppression_for_lochandaka(Mat response, int threshold)
{
	Mat res_img_maxsupp(response.rows, response.cols, CV_8UC1, Scalar(0));
	int value;

	for (int y=1; y<response.rows; y++)
	{
		for (int x=1; x<response.cols; x++)
		{
			//printf("%d,%d-%.2f # ",y, x, res_img_maxsupp.at<float>(y, x));
			//printf("%d,%d-%d \n ",y, x, res_img_maxsupp.at<uchar>(y, x));
			if(response.at<uchar>(y, x) < threshold)
			{
				value = 0;
			}
			else
			{
				value = 255;
			}
			res_img_maxsupp.at<uchar>(y, x) = value;
			//res_img_maxsupp.at<uchar>(y, x) = response.at<uchar>(y, x);
		}
	}

	return res_img_maxsupp;
}
```

Source code of watermark generator

```
int katussa_intensity_diff_value(int intensity_original, int intensity_watermark)
{
	int intensity_diff;
	char *binary;
	binary = (char*)malloc(32+1);

	// Check intensity is < 240.
	if(intensity_original>1 && intensity_original<248)
	{
		// Compeire intensity values of original image and watermark.
		if(intensity_original>=intensity_watermark)
		{

			// LSB of difference is `0` add `1` make odd difference.
			// Difference odd means intensity of orginal image > intensity of watermark.
			binary = katussa_decimal_to_binary(sqrt(intensity_original-
intensity_watermark));
			if(atoi(&binary[31])==0)
			{
				intensity_diff =
katussa_binary_addition(katussa_binary_to_decimal(atoi(binary)),1);

			}
			else{
				intensity_diff =
katussa_binary_addition(katussa_binary_to_decimal(atoi(binary)),0);
```

```
                        }

                }
                else
                {

                        // LSB of difference is `1` add `1` make even difference.
                        // Difference even means intensity of watermark > intensity of original image
                        binary = katussa_decimal_to_binary(sqrt(intensity_watermark-
intensity_original));
                        if(atoi(&binary[31])==0)
                        {
                                intensity_diff =
katussa_binary_addition(katussa_binary_to_decimal(atoi(binary)),0);

                        }
                        else{
                                intensity_diff =
katussa_binary_addition(katussa_binary_to_decimal(atoi(binary)),1);
                        }

                }
        }

        //printf("%s - %s- %d,", binary, &binary[31], intensity_diff);
        free(binary);

        return intensity_diff;
}
```

Source code of embedder

```
/**
 * This function embed a watermark into original image
 * Arguments: int orgimg_width, int orgimg_height (Dimentions of original image)
 * Arguments: int wtmimg_width, int wtmimg_height (Dimentions of watermark image)
 * Arguments: IMG_MATRIX *pixeli_original_image (Pixel values of original image)
 * Arguments: IMG_MATRIX *pixeli_watermark_image (Pixel values of watermark image)
 * This function call several algorithms
 * Return type IMG_MATRIX
 *
*/
IMG_MATRIX *katussa_embed_watermark_corners_lochandaka_single(char *watermark_meta_file_name,
IMG_MATRIX_GRAY *corner_point,

                                                        int orgimg_width, int orgimg_height,

                                                        int wtmimg_width, int wtmimg_height,

                                                        IMG_MATRIX *pixeli_original_image,
IMG_MATRIX *pixeli_watermark_image)
{
```

```
	int index = orgimg_width*orgimg_height*41;
	IMG_MATRIX *img_matrix = NULL;
 img_matrix = (IMG_MATRIX*) malloc(index);

	FILE *watermark_meta_file;
	FILE *watermark_info_file;
	char *watermark_info_file_name;

	// Create watermark meta file
	watermark_meta_file = fopen(watermark_meta_file_name, "w+");

	// Declare the local variables for embed algorithm
	long int total_pixel_image = orgimg_width*orgimg_height;
	int total_pixel_watermark = wtmimg_width*wtmimg_height;
	int x, y;
	int intensity_diff_r, intensity_diff_g, intensity_diff_b;
	//int watermark_embed_index = (total_pixel_image/total_pixel_watermark)-(wtmimg_width/2);

	long int count = 0;
	for(long int i=0; i<total_pixel_image; i++)
	{
		x = i%orgimg_width;
		y = i/orgimg_width;

		if( (y>=corner_point->y && y<corner_point->y+wtmimg_width) && (x>=corner_point-
>x && x<corner_point->x+wtmimg_height) )
		{

			for(int j=0; j<total_pixel_watermark; j++)
			{
				if(count==j)
				{
					intensity_diff_r =
katussa_intensity_diff_value(pixeli_original_image[i].intensity_r, pixeli_watermark_image[j].intensity_r);
					intensity_diff_g =
katussa_intensity_diff_value(pixeli_original_image[i].intensity_g, pixeli_watermark_image[j].intensity_g);
					intensity_diff_b =
katussa_intensity_diff_value(pixeli_original_image[i].intensity_b, pixeli_watermark_image[j].intensity_b);

					img_matrix[i].x = x;
					img_matrix[i].y = y;
					img_matrix[i].intensity_r = pixeli_original_image[i].intensity_r
+ intensity_diff_r;
					img_matrix[i].intensity_g =
pixeli_original_image[i].intensity_g + intensity_diff_g;
					img_matrix[i].intensity_b =
pixeli_original_image[i].intensity_b + intensity_diff_b;

					fprintf(watermark_meta_file, "%d, %d, %d, %d\n", j,
intensity_diff_r, intensity_diff_g, intensity_diff_b);
				}
			}
			count++;
		}
		else
		{

			img_matrix[i].x = x;
			img_matrix[i].y = y;
```

```
                        img_matrix[i].intensity_r = pixeli_original_image[i].intensity_r;
                        img_matrix[i].intensity_g = pixeli_original_image[i].intensity_g;
                        img_matrix[i].intensity_b = pixeli_original_image[i].intensity_b;

                }

        }

        fclose(watermark_meta_file);

        // Create watermark info file
        watermark_info_file_name = (char*)malloc(100);
        strcpy(watermark_info_file_name,  watermark_meta_file_name);
        strcat(watermark_info_file_name, ".info");
        watermark_info_file = fopen(watermark_info_file_name, "w+");
        fprintf(watermark_info_file, "%s, %d, %d, %d, %d\n",
                                        watermark_meta_file_name, orgimg_width, orgimg_height,
wtmimg_width, wtmimg_height);

        return img_matrix;
}
```

Source code of extractor

```
/**
 * This function extract a watermark from watermarked image & watermark meta file
 * Arguments: char* extract method
 * Arguments: char* watermark_meta_file_name
 * Arguments: int orgimg_width, int orgimg_height (Dimentions of original image)
 * Arguments: IMG_MATRIX *pixeli_watermarked_image (Pixel values of watermarked image)
 * Arguments: IMG_MATRIX *pixeli_watermark_meta (Pixel values of watermark meta file)
 * This function call several algorithms
 * Return type IMG_MATRIX
 *
*/
IMG_MATRIX *katussa_extract_watermark_corners_single(IMG_MATRIX_GRAY *corner_point,

                                                int watermarked_width, int watermarked_height,

                                                int wtmimg_width, int wtmimg_height,

                                                IMG_MATRIX *pixeli_watermarked_image, Mat
watermarked_image,

                                                char *watermark_mata_file_name)
{

        int index = wtmimg_width*wtmimg_height*41;
        IMG_MATRIX *img_matrix = NULL;
 img_matrix = (IMG_MATRIX*) malloc(index);

        long int total_pixel_image = watermarked_width*watermarked_height;
```

```
		int total_pixel_watermark = wtmimg_width*wtmimg_height;
		int x_wtm, y_wtm, x_wtmed, y_wtmed; // dimention of watermark image
		int intensity_orgi_r=0, intensity_orgi_g=0, intensity_orgi_b=0;
		int intensity_extrctwtm_r=0, intensity_extrctwtm_g=0, intensity_extrctwtm_b=0;

		// Declare image metrix and allocate memory
		int  index_wtm= total_pixel_watermark*36;
		IMG_DATA *wtm_data = NULL;
 wtm_data = (IMG_DATA*) malloc(index_wtm);

		wtm_data = katussa_read_watermark_meta_data(watermark_mata_file_name);
		/*for(long int i=0; i<total_pixel_watermark; i++)
		{
			printf("%d(%d, %d, %d)# ", wtm_data[i].index, wtm_data[i].intensity_r,
wtm_data[i].intensity_g, wtm_data[i].intensity_b);
		}*/

		long int count = 0;
		for(int i=0; i<total_pixel_watermark; i++)
		{
			x_wtm = i%wtmimg_width;
			y_wtm = i/wtmimg_width;
			//printf("%d,%d# ", img_matrix[i].y, img_matrix[i].y);

			for(int j=corner_point->y+y_wtm; j<corner_point->y+1+y_wtm; j++)
			{
				for(int k=corner_point->x; k<corner_point->x+48; k++)
				{
					img_matrix[i].y = y_wtm;
					img_matrix[i].x = x_wtm;
					//printf("%d,%d# ", j, k);
					//printf("%d,%d# ", img_matrix[i].y, img_matrix[i].x);
					int b = watermarked_image.at<cv::Vec3b>(j, k)[0];
					int g = watermarked_image.at<cv::Vec3b>(j, k)[1];
					int r = watermarked_image.at<cv::Vec3b>(j, k)[2];
					//printf("(%d,%d)-%d \n", j, k, b);
					//img_matrix[i].intensity_r = r;
					//img_matrix[i].intensity_g = g;
					//img_matrix[i].intensity_b = b;

					// If intensity_r odd (if LSB is 1)
					if( (wtm_data[count].intensity_r%2)==1 )
					{
						intensity_orgi_r = r - wtm_data[count].intensity_r;
						intensity_extrctwtm_r = intensity_orgi_r -
(wtm_data[count].intensity_r * wtm_data[count].intensity_r);
						img_matrix[i].intensity_r =
katussa_pixel_intensity_boundary_adjust(intensity_extrctwtm_r);
					}
					// If intensity_g odd (if LSB is 1)
					if( (wtm_data[count].intensity_g%2)==1 )
					{
						intensity_orgi_g = g - wtm_data[count].intensity_g;
						intensity_extrctwtm_g = intensity_orgi_g -
(wtm_data[count].intensity_g * wtm_data[count].intensity_g);
						img_matrix[j].intensity_g =
katussa_pixel_intensity_boundary_adjust(intensity_extrctwtm_g);
					}
					// If intensity_b odd (if LSB is 1)
```

```
                                        if( (wtm_data[count].intensity_b%2)==1 )
                                        {
                                                intensity_orgi_b = b - wtm_data[count].intensity_b;
                                                intensity_extrctwtm_b = intensity_orgi_b -
(wtm_data[count].intensity_b * wtm_data[count].intensity_b);
                                                img_matrix[j].intensity_b =
katussa_pixel_intensity_boundary_adjust(intensity_extrctwtm_b);
                                        }

                                        // If intensity_r odd (if LSB is 0)
                                        if( (wtm_data[count].intensity_r%2)==0 )
                                        {
                                                intensity_orgi_r = r - wtm_data[count].intensity_r;
                                                intensity_extrctwtm_r = intensity_orgi_r +
(wtm_data[count].intensity_r * wtm_data[count].intensity_r);
                                                img_matrix[i].intensity_r =
katussa_pixel_intensity_boundary_adjust(intensity_extrctwtm_r);
                                        }
                                        // If intensity_g odd (if LSB is 0)
                                        if( (wtm_data[count].intensity_g%2)==0 )
                                        {
                                                intensity_orgi_g = g - wtm_data[count].intensity_g;
                                                intensity_extrctwtm_g = intensity_orgi_g +
(wtm_data[count].intensity_g * wtm_data[count].intensity_g);
                                                img_matrix[i].intensity_g =
katussa_pixel_intensity_boundary_adjust(intensity_extrctwtm_g);
                                        }
                                        // If intensity_b odd (if LSB is 0)
                                        if( (wtm_data[count].intensity_b%2)==0 )
                                        {
                                                intensity_orgi_b = b - wtm_data[j].intensity_b;
                                                intensity_extrctwtm_b = intensity_orgi_b +
(wtm_data[count].intensity_b * wtm_data[count].intensity_b);
                                                img_matrix[i].intensity_b =
katussa_pixel_intensity_boundary_adjust(intensity_extrctwtm_b);
                                        }

                                }
                        }

                        //printf("%ld, ", count);
                        count++;
                }

                return img_matrix;
}


/**
 * This function extract a watermark from watermarked image & watermark meta file
 * Arguments: char* extract method
 * Arguments: char* watermark_meta_file_name
 * Arguments: int orgimg_width, int orgimg_height (Dimentions of original image)
 * Arguments: IMG_MATRIX *pixeli_watermarked_image (Pixel values of watermarked image)
 * Arguments: IMG_MATRIX *pixeli_watermark_meta (Pixel values of watermark meta file)
 * This function call several algorithms
 * Return type IMG_MATRIX
 * TODO NEED TO DEVELOPMENT FROMTHE SCRECH
*/
```

```
IMG_MATRIX *katussa_extract_watermark_corners_multiple(IMG_MATRIX_GRAY *corner_point,
                                                       int watermarked_width, int watermarked_height,
                                                       int wtmimg_width, int wtmimg_height,
                                                       IMG_MATRIX *pixeli_watermarked_image, Mat
watermarked_image,
                                                       char *watermark_mata_file_name)
{
        int index = wtmimg_width*wtmimg_height*41;
        IMG_MATRIX *img_matrix = NULL;
 img_matrix = (IMG_MATRIX*) malloc(index);

        long int total_pixel_image = watermarked_width*watermarked_height;
        int total_pixel_watermark = wtmimg_width*wtmimg_height;
        int x_wtm, y_wtm, x_wtmed, y_wtmed; // dimention of watermark image
        int intensity_orgi_r=0, intensity_orgi_g=0, intensity_orgi_b=0;
        int intensity_extrctwtm_r=0, intensity_extrctwtm_g=0, intensity_extrctwtm_b=0;

        // Declare image metrix and allocate memory
        int  index_wtm= total_pixel_watermark*36;
        IMG_DATA *wtm_data = NULL;
 wtm_data = (IMG_DATA*) malloc(index_wtm);

        wtm_data = katussa_read_watermark_meta_data(watermark_mata_file_name);
        /*for(long int i=0; i<total_pixel_watermark; i++)
        {
                printf("%d(%d, %d, %d)# ", wtm_data[i].index, wtm_data[i].intensity_r,
wtm_data[i].intensity_g, wtm_data[i].intensity_b);
        }*/

        long int count = 0;
        for(int i=0; i<total_pixel_watermark; i++)
        {
                x_wtm = i%wtmimg_width;
                y_wtm = i/wtmimg_width;
                //printf("%d,%d# ", img_matrix[i].y, img_matrix[i].y);

                for(int j=corner_point->y+y_wtm; j<corner_point->y+1+y_wtm; j++)
                {
                        for(int k=corner_point->x; k<corner_point->x+48; k++)
                        {
                                img_matrix[i].y = y_wtm;
                                img_matrix[i].x = x_wtm;
                                //printf("%d,%d# ", j, k);
                                //printf("%d,%d# ", img_matrix[i].y, img_matrix[i].x);
                                int b = watermarked_image.at<cv::Vec3b>(j, k)[0];
                                int g = watermarked_image.at<cv::Vec3b>(j, k)[1];
                                int r = watermarked_image.at<cv::Vec3b>(j, k)[2];
                                //printf("(%d,%d)-%d \n", j, k, b);
                                //img_matrix[i].intensity_r = r;
```

```
//img_matrix[i].intensity_g = g;
//img_matrix[i].intensity_b = b;

// If intensity_r odd (if LSB is 1)
if( (wtm_data[count].intensity_r%2)==1 )
{
	intensity_orgi_r = r - wtm_data[count].intensity_r;
	intensity_extrctwtm_r = intensity_orgi_r -
(wtm_data[count].intensity_r * wtm_data[count].intensity_r);
	img_matrix[i].intensity_r =
katussa_pixel_intensity_boundary_adjust(intensity_extrctwtm_r);
}
// If intensity_g odd (if LSB is 1)
if( (wtm_data[count].intensity_g%2)==1 )
{
	intensity_orgi_g = g - wtm_data[count].intensity_g;
	intensity_extrctwtm_g = intensity_orgi_g -
(wtm_data[count].intensity_g * wtm_data[count].intensity_g);
	img_matrix[j].intensity_g =
katussa_pixel_intensity_boundary_adjust(intensity_extrctwtm_g);
}
// If intensity_b odd (if LSB is 1)
if( (wtm_data[count].intensity_b%2)==1 )
{
	intensity_orgi_b = b - wtm_data[count].intensity_b;
	intensity_extrctwtm_b = intensity_orgi_b -
(wtm_data[count].intensity_b * wtm_data[count].intensity_b);
	img_matrix[j].intensity_b =
katussa_pixel_intensity_boundary_adjust(intensity_extrctwtm_b);
}

// If intensity_r odd (if LSB is 0)
if( (wtm_data[count].intensity_r%2)==0 )
{
	intensity_orgi_r = r - wtm_data[count].intensity_r;
	intensity_extrctwtm_r = intensity_orgi_r +
(wtm_data[count].intensity_r * wtm_data[count].intensity_r);
	img_matrix[i].intensity_r =
katussa_pixel_intensity_boundary_adjust(intensity_extrctwtm_r);
}
// If intensity_g odd (if LSB is 0)
if( (wtm_data[count].intensity_g%2)==0 )
{
	intensity_orgi_g = g - wtm_data[count].intensity_g;
	intensity_extrctwtm_g = intensity_orgi_g +
(wtm_data[count].intensity_g * wtm_data[count].intensity_g);
	img_matrix[i].intensity_g =
katussa_pixel_intensity_boundary_adjust(intensity_extrctwtm_g);
}
// If intensity_b odd (if LSB is 0)
if( (wtm_data[count].intensity_b%2)==0 )
{
	intensity_orgi_b = b - wtm_data[j].intensity_b;
	intensity_extrctwtm_b = intensity_orgi_b +
(wtm_data[count].intensity_b * wtm_data[count].intensity_b);
	img_matrix[i].intensity_b =
katussa_pixel_intensity_boundary_adjust(intensity_extrctwtm_b);
}
```

```
                        }
                }

                //printf("%ld, ", count);
                count++;
        }

        return img_matrix;
}
```