M. Tech (Signal Processing & Digital Design) PRANEESH GUPTA 2020

THREE PHASE DIGITAL IMAGE REVERSIBLE WATERMARKING WITH LOCALIZED TAMPER DETECTION

A DISSERTATION

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE
OF

MASTER OF TECHNOLOGY
IN
SIGNAL PROCESSING AND DIGITAL DESIGN

Submitted by:

PRANEESH GUPTA
2K18/SPD/11

Under the supervision of

Dr. Jeebananda Panda
Professor, DTU

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

AUGUST, 2020

THREE PHASE DIGITAL IMAGE REVERSIBLE WATERMARKING WITH LOCALIZED TAMPER DETECTION

A PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE
OF

MASTER OF TECHNOLOGY
IN
SIGNAL PROCESSING AND DIGITAL DESIGN

Submitted by

PRANEESH GUPTA
2K18/SPD/11

Under the supervision of

Dr. Jeebananda Panda
Professor, DTU

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

AUGUST, 2020

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I **Praneesh Gupta (Roll No. 2K18/SPD/11),** student of M.Tech (Signal Processing and Digital Design), hereby declare that the Project Dissertation titled **"Three Phase Digital Image Reversible Watermarking with Localized Tamper Detection"** which is submitted by me to the Department of Electronics and Communication Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. To the best of my knowledge, this work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Praneesh Gupta.

Place: Delhi **(PRANEESH GUPTA)**

Date: 28th August 2020

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CERTIFICATE

I hereby certify that the Project Dissertation titled "**Three Phase Digital Image Reversible Watermarking with Localized Tamper Detection**" which is submitted by **Mr. Praneesh Gupta (Roll No. 2K18/SPD/11),** Department of Electronics and Communication, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the student under my supervision. To the best of my knowledge, this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi

Date:

(Dr. Jeebananda Panda)
SUPERVISOR
Professor, ECE Department
Delhi Technological University

ACKNOWLEDGEMENT

I would like to thank Defence Research and Development Organisation (DRDO) for giving me this opportunity to foster my academic interest.

I would like to express my gratitude towards all the people who have contributed their precious time and effort to help me without whom it would not have been possible for me to understand and complete the project.

I would like to thank Prof. Jeebananda Panda, Department of Electronics and Communication Engineering, my Project guide, support, motivation and encouragement throughout the period this work was carried out. His readiness for consultation at all times, his educative comments, his concern and assistance even with practical things have been invaluable.

I am also thankful to Dr. P R Mishra, Scientist, SAG DRDO and Ms. Lavi Tanwar, Assistant Professor, ECE Department, Delhi Technological University for their timely advice and help which was a constant source of inspiration.

(PRANEESH GUPTA)

ABSTRACT

Reversible watermarking is a lossless watermarking technique, which allows the extraction of watermark as well as the recovery of original cover media at the receiver end for ensuring authentication and integrity verification of digital data. In this project, a high capacity improved median based three phase reversible watermarking algorithm using prediction error expansion for grayscale digital images is proposed. The scheme predicts the candidate pixels using median of original neighboring pixels, which keeps magnitude of prediction errors relatively small, also the prediction errors in particular phase are sorted according to context pixels variance to obtain low distortion for small payloads. Furthermore, I have shown the efficient creation of location map that requires only single bit to test for overflow/underflow. The performance of the scheme is analyzed for some of the publicly available standard and medical test images in terms of maximum embedding capability, PSNR, SSIM index values and location map size. The scheme is also compared with some of the state-of-the-art schemes. The comparison results demonstrate that the proposed scheme has lesser distortion for different payload.

Further, in this project I have also presented localized tamper detection in the received digital image as one of the application of the proposed scheme which allows the receiver to reject only the selective regions in case of integrity failure. In this application, the image is divided into blocks, the integrity value is then computed taking into account all the block pixels and the corresponding block number, subsequently this value is reversibly embedded in the block. The current and next blocks are combined dynamically when embedding capacity of a block is not sufficient. The experimental results on some test images indicate that the watermarked image has high PSNR and SSIM value also the results under various attacks demonstrates the effectiveness of the algorithm.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS, ABBREVIATIONS

Abbreviation	Full form
DE	Difference Expansion
EC	Embedding Capacity
FRR	False Rejection Rate
LSB	Least Significant Bit
MAC	Message Authentication Code
MSE	Mean Squared Error
PEE	Prediction Error Expansion
PEH	Prediction Error Histogram
PSNR	Peak Signal-to-Noise Ratio
SHA	Secure Hash Algorithm
SSIM	Structural Similarity

CHAPTER 1

INTRODUCTION

1.1. Preface

Digital communication have a very important role in the modern world. Information security has a key role in communication for securing our sensitive data. Typically, the information security systems can be classified into cryptography and information hiding. Fig 1.1 shows the classification of security systems. Both cryptography and information hiding systems are employed for securing the sensitive information but the approach used for securing the sensitive data is entirely different.

Cryptography converts the plaintext into cipher text by using a secret encryption key and an encryption algorithm, the receiver on the other side decrypt the cipher text back to plain text by using the secret decryption key and decryption algorithm. Information hiding of digital data can be achieved using steganography and watermarking. Steganography is used to hide the secret message within another digital media known as cover, by changing some of the properties of the cover data.

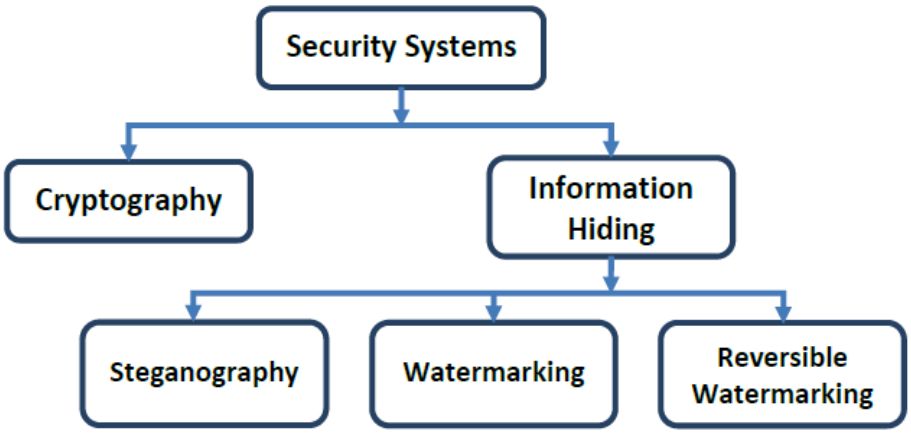


Figure 1.1: Classification of Security Systems

Digital watermarking is a mechanism in which some digital data (watermark) is embedded in another digital media, which may be audio, video or image with an aim to achieve integrity/authentication detection or protection of copyright. The watermarking technique is known as irreversible when the complete recovery of cover data is not possible and hence it leads to a permanent modification in the original cover. To avoid this shortcoming, reversible watermarking is introduced in which the extraction of watermark together with lossless restoration of the original host data is possible.

1.2. Objectives and Problem Definition

Information hiding is a process which is used to embed some digital data into another digital media which may be image, text, video or audio. Reversible watermarking or reversible data hiding is relatively a new field in information hiding, in which the watermark as well as the complete cover image is recovered at the receiving end. The objectives of this project report are:

(i) To develop a high capacity with low distortion reversible watermarking embedding and extraction algorithm based on PEE (prediction error expansion) and shifting of prediction error histogram.
(ii) To analyze the performance of the algorithm for different test images in terms of maximum embedding capability, watermark imperceptibility using PSNR, structural degradation using SSIM index and length of location map.
(iii) Implementation of the application of proposed algorithm for the purpose of localized tamper detection in the received/stored digital image.

1.3. Organization of the Project Report

Chapter 2 represents the overview of the digital watermarking technology, its trade-offs, classification, comparison of different watermarking techniques and various applications.

Chapter 3 represents the overview of reversible watermarking technology, its classification, performance evaluation criterion and various applications of reversible watermarking.

Chapter 4 represents the detail description of the proposed methodology of reversible watermarking algorithm using improved median based prediction. It includes description of prediction of candidate pixels, watermark embedding & generation of location map and process for watermark extraction and restoration of cover image.

Chapter 5 represents detail description for implementation of localized tamper detection algorithm in the received image using proposed reversible watermarking scheme. This chapter also discusses selection criterion of integrity check algorithm for calculating the integrity of blocks based on different conditions and attack scenario.

Chapter 6 represents the implementation results of the proposed reversible watermarking and localized tamper detection algorithm. It also contains the comparison results of the proposed method with some of the state-of-the-art reversible watermarking schemes.

Finally, chapter 7 will discuss the results; draw conclusions and discuss about future scope.

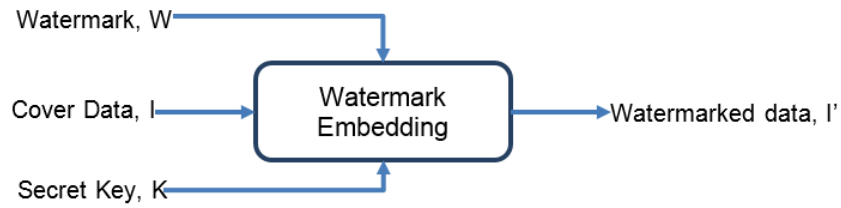
CHAPTER 2

DIGITAL WATERMARKING

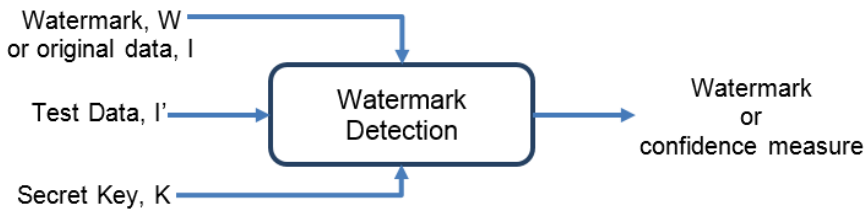
This chapter presents an overview of digital watermarking technology, its trade-offs, classification, comparison of different watermarking techniques and various applications.

2.1. Introduction

Digital watermarking is a scheme that inserts some digital data (termed as digital watermark) into another digital media viz. text, image, audio etc. The digital information can be author's name, serial number, text, logo of company, etc. The embedded watermark in the digital media can be visible or invisible depend on the application. Digital watermarking is used to provide evidence of ownership, tamper verification of digital data, etc. Fig 2.1 represents the general framework of watermark embedding and its detection [1].



(a) Watermark Embedding



(b) Watermark Detection

Figure 2.1: Watermark Embedding and Detection Process

Digital watermark embedding is carried either in spatial or in frequency domain, each have their own pros and cons. In spatial-domain watermarking, the watermark is inserted into image by straightway changing some of the selected pixels. The spatial domain watermarking is simple and it's embedding and extraction speed is higher than frequency domain but in general, the spatial domain watermarking is less robust against attacks. Examples of spatial domain watermarking are LSB, additive watermarking etc.

In transform domain based digital watermarking, the information is embedded by tweaking some of the frequency coefficients of host image, then inverse transform of altered frequency coefficients is taken to obtain the marked image [2]. Some of the transforms used for watermarking are DFT, DCT and DWT. Fig. 2.2 shows the comparison of spatial and transform domain techniques [3].

	Spatial Domain	Frequency Domain
Complexity	Low	High
Robustness	Fragile	Semi Fragile, Robust
Image Quality	High Control	Low Control
Embedding Capacity	High (in general)	Low
Usage	Integrity, Authentication	Copyright Protection

Figure 2.2: Comparison for Spatial Domain and Frequency Domain

Comparison of different techniques employed for watermark embedding in spatial and transform domain are given in Table below [4] [5].

Table I: Comparison of Different Watermarking Techniques

Algorithm	Advantages	Disadvantages
Least Significant Bit	1. Simple and less complex. 2. Perceptual quality of the marked image is high.	1. Low robustness. 2. Sensitive to attack and noise. 3. Sensitive to scaling, rotation etc.
DCT (Discrete Cosine Transform) based	Middle frequency coefficients are utilized to embed the watermark, which results in good quality marked image and provide robustness against many image-processing operations.	1. Invariance properties will get destroyed with the block wise computation of DCT. 2. During quantization step, some of the higher frequency components will get suppressed.

DWT (Discrete Wavelet Transform) based	1. DWT presents good localization in both spatial and frequency domains. 2. Compression ratio is high that is related to human visual system.	1. Computation cost is high. 2. Longer compression time.
Discrete Fourier Transform (DFT) based	DFT based watermark sustain with geometric distortions, since it is invariant to rotation, scaling and translation.	1. Complex implementation 2. Round off errors

2.2. Watermarking Trade-Offs

The primary requirements of watermarking algorithms are EC (embedding capacity), robustness and marked image quality as shown in Fig. 2.3 [6].

Quality: Error introduced due to embedding in the cover image. The lower the distortion of the cover data after watermarking, the higher is the perceptual quality.

Capacity: Maximum size of watermark bits (without side information) that is possible to embed in single layer into a cover image by watermark embedding algorithm.

Robustness: The ability of the watermarked image to withstand against different types of modification or attack. If the watermarking scheme is more robust, then there is lower probability that a modification in the watermarked image being detected at receiver.

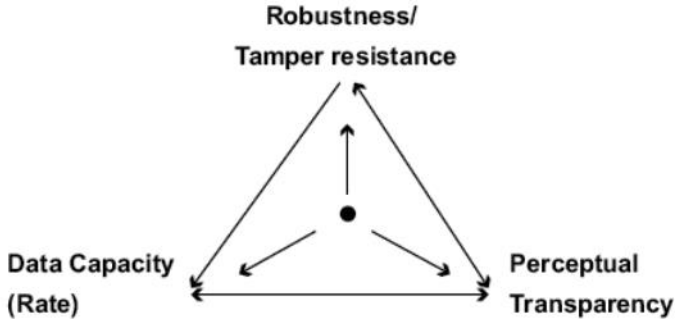


Figure 2.3: Primary Requirements of Watermarking Algorithms

If we try to improve one of the parameters (perceptual quality, data capacity or robustness) for a particular watermarking scheme then it usually deteriorates one or both of the others. We have to attain a trade-off between maximum embedding capacity, robustness and perceptual quality when we have to design a watermarking algorithm, so that it can be used in different applications. In other words, watermarking scheme must

be designed in such a way that the performance of the scheme should lie somewhere within the triangle shown in Fig. 2.3.

2.3. Classification of Digital Watermarking

Digital watermarking embeds digital information or code into cover image, audio or video cover which may be visible or invisible. The digital watermarking methods can be classified as shown in Fig. 2.4 [7].

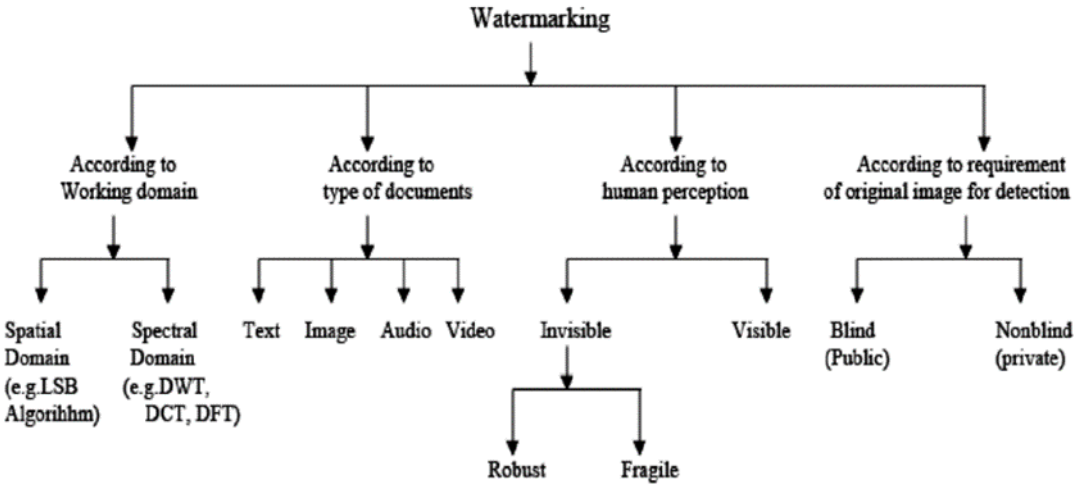


Figure 2.4: Classification of Watermarking

2.4. Applications of Digital Watermarks

Digital watermarking techniques hides information (copyright, authorized recipient's information or integrity value) into the cover data. As compared to cryptographic systems, digital watermarking represents an efficient technique to ensure source authentication as well as integrity verification. Applications of digital watermarking techniques is categorized into following five major classes [6, 10]:

a) Copyright Protection
b) Fingerprinting
c) Integrity Protection
d) Copy Control
e) Device Control

CHAPTER 3

LITERATURE REVIEW

This chapter presents an overview of reversible watermarking technology, its classification, performance evaluation criterion and various applications of reversible watermarking.

3.1. Introduction

Digital watermarking technique is helpful in proving the ownership of the data through identification of watermark, but the drawback is that it introduces modification in the pixels of cover image, which are irreversible in nature, and causes the resulting recovered image after watermark extraction to become different from the original cover image. To avoid this shortcoming, reversible watermarking is introduced which permits the extraction of watermark together with lossless restoration of complete cover data.

Reversible watermarking (also known as *lossless, invertible or reversible data hiding)* is an emerging and relatively new area and it is used for providing copyright protection together with verification of integrity and source authentication of received data [8]. It has applications in sensitive industries like medical industries, military imaging, forensics industries etc. where complete recovery of sensitive cover data is highly important and distortion in the data is difficult to be tolerated, even if not perceptually significant. In such sensitive applications, *reversible watermarking* schemes are very useful as these algorithms permits the lossless retrieval of the original cover data content. Reversible watermarking algorithms are in general belong to fragile class of watermarking. Fig. 3.1 depicts the block diagram of general reversible watermarking system.

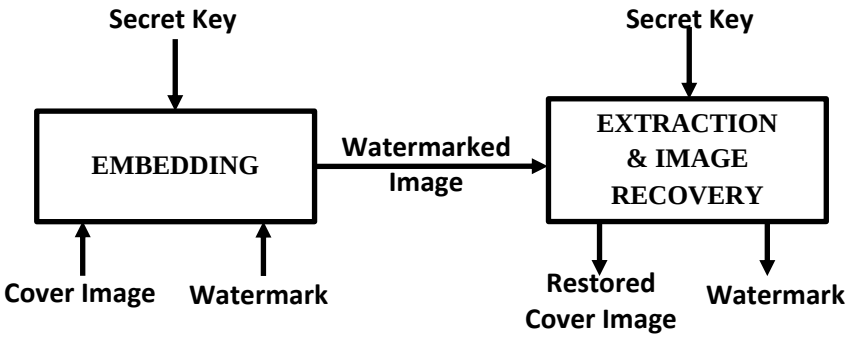


Figure 3.1: Basic Reversible Watermarking Scheme

Generally, in many reversible data hiding applications, for example, for the purpose of integrity verification of digital data, the watermark which is embedded is a concatenation of integrity value (calculated using cryptographic hash function for e.g. SHA or MD5) of an original cover and the actual information (payload) [6]. This embedded watermark permits the recipient to authenticate or verify the integrity of the watermarked image without the requirement of additional information (cover data). In order to attain both integrity and source authentication, message authentication code (MAC) which uses secret key for integrity value calculation will be embedded as a watermark.

3.2. Classification of Reversible Watermarking

Reversible watermarking algorithms based on robustness can be classified into fragile and semi-fragile algorithms. Reversible watermarking techniques is in general a fragile watermarking technique as per literature i.e. in case of any attack or image processing application on the watermarked image, the exact watermark as well as the revival of actual cover image is not possible. Reversible watermarking algorithms belonging to semi-fragile class are very less in number in comparison to fragile class. Semi fragile class of algorithms are designed in such a way that the watermark can be retrieved even with some unintentional changes in the watermarked data like small distortion or slight image compression [8].

Reversible watermarking schemes can be mainly classified based on the techniques used for embedding and extraction into following types [9] [10]:

- Histogram-Bin-Shifting based
- Lossless Data Compression based

- ❖ Difference Expansion (DE) based
- ❖ Prediction Error Expansion (PEE) based
- ❖ Transform domain based

3.2.1. Histogram Bin Shifting based Reversible Watermarking

Reversible watermarking algorithms belongs to histogram-bin shifting class embed watermark bits in the cover using image histogram (i.e. frequency of occurrence of pixel intensity values). The different reversible watermarking schemes employing this method are introduced in [11] [12].

For example, in the algorithm presented by Ni et. al. [11], for embedding of watermark bit some pair of peak point and minimum points are selected. The bin in the image histogram immediately next to the maximum or peak bin is emptied by shifting the bins right. Then the image is scanned, if the pixel $x = peak\ value$ is found, then embedding is done by adding the watermark bit $w \in \{0,1\}$ to the peak pixel value i.e. $x = x + w$. The drawback of this strategy is that the EC is limited to the total pixels with maximum intensities. Fig. 3.2(a) and (b) shows the histograms of Lena image before and after embedding watermark. [11]

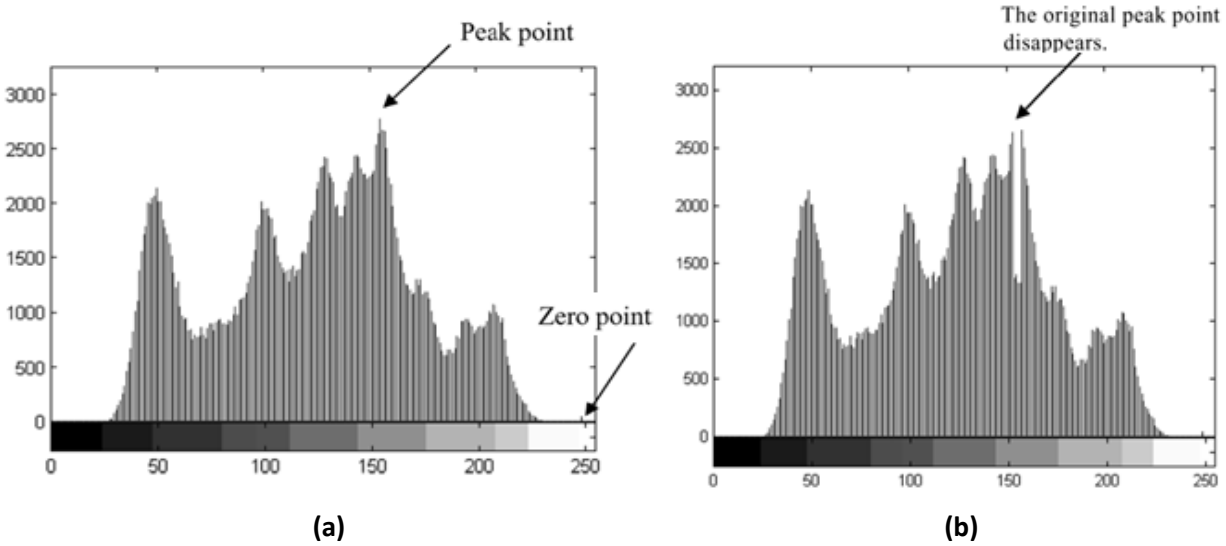
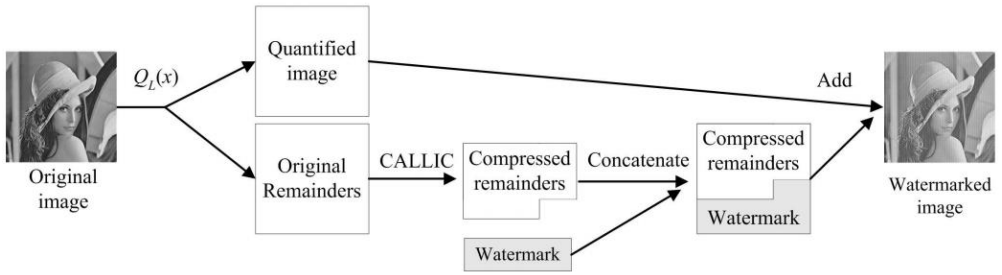


Figure 3.2: Histogram of (a) Cover and (b) Watermarked Lena Image (512 x 512)

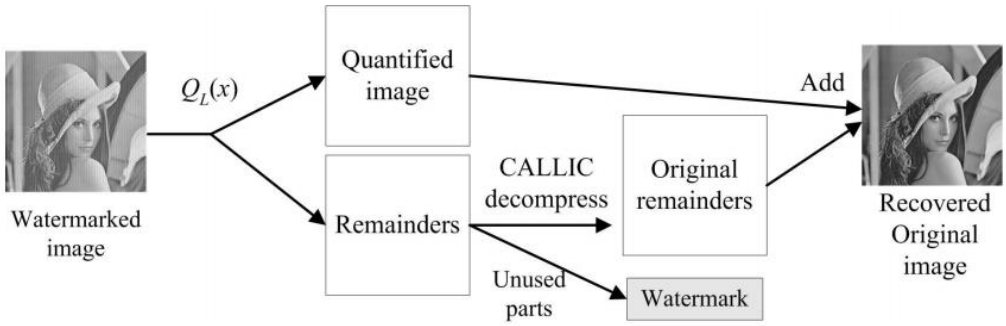
3.2.2. Lossless Data Compression based Reversible Watermarking

Reversible watermarking schemes belongs to this class [13] [14] [15], compresses some information of the cover image like some bit planes, to make extra space to embed bits in the image. In general, to achieve less distortion in the watermarked image, the extra space is created by compressing the lower bit planes of the cover image. Xuan et al. [14] presented a reversible watermarking scheme that compresses some of the integer wavelet coefficients; the companding function used in the scheme compresses the coefficients that are greater than certain threshold. This leads to increase in EC but the overhead also get increased. The threshold technique was modified by Memon et al. [15] to enhance the embedding capacity.

In Celik et al.'s [13] reversible watermarking algorithm, for embedding watermark into L lowest bit planes; the cover image pixels are quantized into L-levels and the watermark bits are mapped to L-ary symbols. The remainder array is then losslessy compressed and combined with symbol array to get concatenated array. The watermarked image is obtained by adding the quantized image with concatenated array. The embedding and extraction process of [13] is shown in Fig. 3.3 [16].



(a) Embedding Process



(b) Extraction and Image Recovery Process

Figure 3.3: Embedding and Extraction Process of Celik et al. Scheme

3.2.3. Difference Expansion (DE) based Reversible Watermarking

Difference expansion (DE) based algorithms exploits the high redundancy in digital images. Tian [17] proposed first algorithm based on difference expansion (DE) in 2003. After that, several reversible watermarking schemes with modified difference expansion were introduced by authors in [18] [19]. In the algorithm proposed by Tian [17], the difference between consecutive pixels are computed to embed the watermark bit. For a pixel pair (x, y), $0 \leq x, y \leq 255$ the average l and difference h is defined as:

$$l = \left\lfloor \frac{x+y}{2} \right\rfloor \text{ and } h = x - y \tag{3.1}$$

The inverse transform is defined as

$$x = l + \left\lfloor \frac{h+1}{2} \right\rfloor \text{ and } y = l - \left\lfloor \frac{h}{2} \right\rfloor \tag{3.2}$$

Since, $0 \leq x, y \leq 255$, we have

$$\begin{gathered} 0 \leq l + \left\lfloor \frac{h+1}{2} \right\rfloor \leq 255 \text{ and } 0 \leq l - \left\lfloor \frac{h}{2} \right\rfloor \leq 255 \\ \text{or, } |h| \leq \min(2(255 - l), 2l + 1) \end{gathered} \tag{3.3}$$

Expandable: The pixel pair (x, y), the difference h, is expandable if

$$|2h + b| \leq \min(2(255 - l), 2l + 1), for\ b = 0\ and\ 1 \tag{3.4}$$

Changeable: The pixel pair (x, y), the difference h, is changeable if

$$\left| \left\lfloor \frac{h}{2} \right\rfloor . 2 + b \right| \leq \min(2(255 - l), 2l + 1)\ , for\ b = 0\ and\ 1 \tag{3.5}$$

Not Changeable: The differences (h), which are not changeable, are also non-expandable.

In order to prevent underflow and overflow conditions, the modified difference number, h' must follow equation (3.3). The pixel pairs that produces underflow/overflow i.e. the pixel pairs constructed after inverse transform say $(x', y') \notin [0,255]$ (for $unsigned\ 8\ bit$ pixels) were not used for embedding and kept intact.

3.2.4. Prediction Error Expansion (PEE) based Reversible Watermarking

Prediction error expansion (PEE) based reversible watermarking schemes has generally high EC (embedding capacity) with low degradation in the marked image. PEE based algorithms uses the correlation present among neighboring pixels in digital images. In this method, at specified locations the pixel values are predicted from nearby pixels

using various statistical methods such as median, rhombus, mean, median edge detector etc. [6] [20] [21] [22]. In this technique, watermark embedding carried out by expanding some of the bins in prediction error histogram (PEH). PEH is generated by computing the difference between the actual candidate pixel and its predicted value. The requirement of state-of-the-art techniques is to reduce the location map size that is used to track the overflow/underflow pixels during embedding, since for blind reversible watermarking location map is also inserted in the watermarked image.

Prediction of pixels at specified location can also be carried out using interpolation method. In this technique, a high-resolution interpolated image is generated from the given low resolution image. Interpolation technique estimates the missing pixels of an image. Reversible algorithm based on interpolation technique exhibits less computational cost and low distortion as it considers the interpolation error for data embedding and for expanding the error it uses addition operation instead of bit shifting [22] [23] [24]. The different types of reversible data hiding methods based on interpolation that are being used commonly are "Neighbour Mean Interpolation (NMI)", "Interpolation by Neighbouring Pixels (INP)" and "Interpolation by Maximizing the Difference Values between Neighbouring Pixels" [23].

Naskar et. al. [20] introduced PEE based reversible watermarking algorithm that utilizes the existing correlation between neighboring pixels, which is in general very high in most of the digital images. During embedding, the cover image pixels based on position categorized into base pixels and three sets namely first set, second set and third set of pixels. The base pixels are treated as reference and hence retain unmodified while other sets of pixels are utilized for embedding. Prediction of the candidate pixels in three sets are carried out in a successive manner, using median of the neighbouring pixels with more weights are provided to the base pixels. Next, the prediction error are computed for candidate pixels of all sets using actual candidate pixels. For embedding a watermark bit, an error threshold (k) is selected, and the prediction error which is less than or equal to threshold level (k) are used to embed the watermark bit while other error magnitudes are shifted by $(k+1)$.

Let x and x'represents the original and predicted candidate pixel, then the error in predicted value is calculated using equation (3.6) as,

$$e = x' - x \tag{3.6}$$

Let e' represents the prediction error modified either by embedding watermark bit $w \in \{0,1\}$ or by shifting, then e' is given as,

$$e' = \begin{cases} sign(e) * (2|e| + w) & if\ |e| \leq k \\ sign(e) * (|e| + k + 1)\ if\ |e| > k \end{cases} \quad (3.7)$$

Let P_{wat} represents the watermarked pixel, then P_{wat} is obtained by subtracting the expanded error e' from the corresponding predicted value x' as,

$$P_{wat} = x' - e' \quad (3.8)$$

The flowchart for embedding watermark bit into prediction errors and generation of watermarked image is depicted in Fig. 3.4.

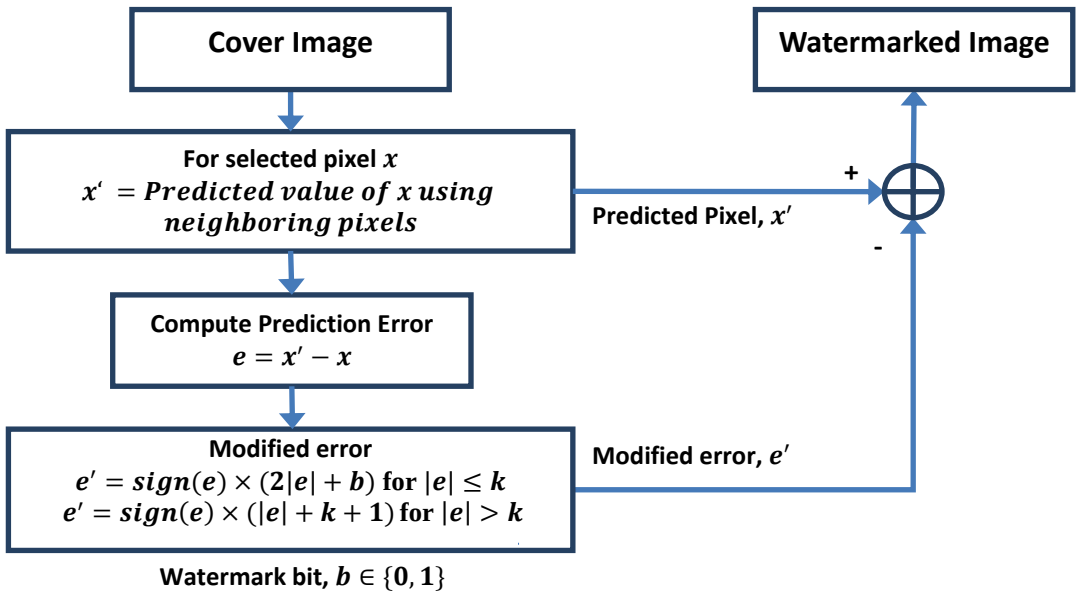


Figure 3.4: Watermark Embedding by Expanding Prediction Error

During embedding, some of the watermarked pixels may produce overflow or underflow i.e. $P_{wat} \notin [0, 255]$, these type of candidate pixels are not transformed i.e., kept unmodified in image and the track of their positions are placed in location map. For blind extraction, the error threshold k together with location map is inserted in the LSBs of base pixels starting from end. To achieve lossless recovery of the image, the original base pixel least-significant bits (LSB's) (used to hide k and location map) are concatenated with the actual watermark to form effective payload.

During extraction and image recovery, the watermarked image pixels are categorized into base pixels and three sets of candidate pixels similar to embedding process. Next, from LSB's of base pixels, overhead information are extracted to obtain

threshold and location map. After obtaining the location map and error threshold, the candidate pixels are predicted and the modified prediction errors are calculated using watermarked pixels. Then test pixels are created and checked for possible overflow/underflow, if occurred then location map are used to obtain the watermark (if bit is embedded) and the original prediction error values are found using equation (3.9) and (3.10).

$$w = e'(mod\ 2)\ \ if\ |e'| \leq (2k+1) \tag{3.9}$$

$$e = \begin{cases} sign(e') * \dfrac{|e'| - b}{2} & if|e'| \leq (2k+1) \\ sign(e') * (e' - k - 1) & if\ \ |e'| > 2k+1 \end{cases} \tag{3.10}$$

After finding the actual prediction error e for each watermarked pixel, the cover image pixels found by using equation (3.11) as,

$$x = x' + e \tag{3.11}$$

3.2.5. Transform Domain based Reversible Watermarking

In transform domain based reversible watermarking schemes, some of the host image coefficients computed using DCT, DFT, integer DWT etc. are altered to embed the watermark [25] [26] [27].

For example, Huang et. al. [26] introduced a reversible watermarking in compressed JPEG images by modifying some of the quantified DCT (Discrete Cosine Transform) coefficients. In order to obtain less distortion in the watermarked image, the embedding is carried out block-by-block and starts with blocks which have more zero coefficients. The forward DCT coefficients for an 8×8 block is given by,

$$F(u,v) = \frac{1}{4}c(u)c(v)\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y)cos\frac{(2x+1)u\pi}{16}\ cos\frac{(2y+1)v\pi}{16} \tag{3.12}$$

Where,

$$c(u) = \begin{cases} 1/\sqrt{2} & if\ u = 0 \\ 1 & otherwise \end{cases}$$

Then, the quantified integer coefficients are obtained as,

$$d(u,v) = round\left(\frac{F(u,v)}{q(u,v)}\right) \tag{3.13}$$

Where, $F(u,v)$ and $q(u,v)$ are the original DCT coefficients and the corresponding step size in quantization table.

The watermark bits $b \in \{0,1\}$ are embedded in the non-zero AC coefficients at bins $1\ and\ -1$ while other bins are shifted to avoid overlapping i.e., the quantified DCT coefficients are modified as,

$$d^w = \begin{cases} d + sign(d) * b\ if\ |d| = 1 \\ d + sign(d) \quad if\ |d| > 1 \end{cases} \tag{3.14}$$

Where,

$$sign(d) = \begin{cases} 1 \ \ if\ d > 0 \\ -1\ if\ d < 0 \\ 0 \ \ if\ d = 0 \end{cases}$$

After embedding complete watermark bits in the quantified coefficients, the coefficients are restored to their actual positions followed by entropy encoding of the modified coefficients to generate the watermarked JPEG file.

3.3. Comparison of Reversible Watermarking Techniques

Most of the reversible watermarking algorithms are fragile in nature and are usually employed for authentication and integrity check in sensitive applications since after successful verification; the complete cover image is restored. Comparison of some algorithms employing different techniques is given in Table below.

Table II: Comparison of some Reversible Watermarking Techniques

Reversible Watermarking Technique	Example	Advantages	Disadvantages
Histogram-Bin-Shifting	Ni et. al. [11]	• Preserves image quality • Low Complexity • Produces small amount of side information	• Embedding capacity limited to number of peak pixels
Lossless Data Compression	Celik et. al. [13]	• Embedding capacity better than Ni et. al. [11]	• Embedding capacity depends on quantization levels • Adds complexity due to lossless image compression algorithm to make space for embedding payload

Difference Expansion (DE)	J. Tian [17]	• High Embedding Capacity • Better capacity vs distortion than Celik et al.	• Relatively high complexity than histogram bin shifting • Large and variable side information
Prediction Error Expansion (PEE)	Naskar et. al. [20]	• High Embedding Capacity • Very small side information • Good PSNR	• Prediction is highly depend on image • Size of location map is of variable size
Transform Domain	Huang et. al. [26]	• Embedding done in compressed JPEG image • Low distortion	• Low embedding capacity

3.4. Performance Evaluation of Reversible Watermarking

The performance of the reversible watermarking algorithm is evaluated with respect to the following properties [6] [20]:

3.4.1. Maximum Embedding Capacity

Maximum embedding capacity (EC) of the reversible watermarking algorithm is evaluated in terms of maximum size of actual watermark (without considering side information required for extraction) that can be embedded into the given host image. It can also be evaluated in terms of average bits that can be embedded in a pixel (given in terms of bpp or bits per pixel).

$$Bits\ per\ pixel\ (bpp) = \frac{EC(bits)}{m * n} \tag{3.15}$$

Where,

$EC = Embedding\ Capacity\ in\ bits$

$m, n = number\ of\ rows\ and\ column\ for\ the\ given\ image$

3.4.2. Distortion Level of Watermarked Image

Distortion level of watermarked image after embedding in the given cover image can be assessed using PSNR (Peak Signal-to-Noise Ratio). For PSNR calculation, mean-squared error (MSE) is required and calculated using equation (3.16).

$$MSE = \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{[P(i,j) - P_{wm}(i,j)]^2}{mn} \tag{3.16}$$

Where, $P(i,j)$ & $P_{wm}(i,j)$ represents cover and watermarked pixel intensity value at $(i,j)^{th}$ position, m & n represents size of given image. Then PSNR is calculated as,

$$PSNR = 10\log_{10}\left(\frac{P_{MAX}^2}{MSE}\right) dB = 10\log_{10}\left(\frac{255^2}{MSE}\right) dB \tag{3.17}$$

Where, P_{MAX} is the largest feasible value of the pixel intensity for the given image [P_{MAX} is taken as 255 for grayscale image with 8 bit pixels].

3.4.3. Structural Similarity (SSIM) Index

The structural similarity index (SSIM) is used to evaluate the degradation in structural information due to watermark embedding, this is based on HVS (human visual system) that extracts the structural information. SSIM index provides the amount of change between images in perceptual terms. In other words, SSIM index compares the pixel intensity patterns locally which is normalized for contrast and luminance. SSIM index gives a better indication of image quality that is perceived visually by humans.

Unlike MSE or PSNR, that estimate *absolute errors* between two images, SSIM is based on visible structures in the image that considers *perceived degradation in structural information* in an image. As an example, the peak signal to noise ratio for a blurred image (in comparison to unblurred image) may be quite high but has low perceived quality.

SSIM index value provides the human visual quality measure of an image that estimates the impact in three properties of an image namely luminance, contrast and structure. The SSIM index value between two $x\ and\ y$ images is given as the multiplicative combination of the three terms as, [28] [29]

$$SSIM(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma} \tag{3.18}$$

Where, $l(x,y), c(x,y) and\ s(x,y)$ represents the luminance, contrast and structure comparison measures respectively, given by

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \tag{3.19}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \tag{3.20}$$

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \tag{3.21}$$

Where μ_x, μ_y are the local means; σ_x, σ_y are the standard deviations; and σ_{xy} is the cross-covariance for images x, y. $C_1, C_2 and\ C_3$ are constants, defined as,

$$C_1 = (K_1L)^2, C_2 = (K_2L)^2\ and\ C_3 = C_2/2$$

Where, L is taken as 255 for unsigned 8 bit integer grayscale image, $K_1 \ll 1\ and\ K_2 \ll 1$ are constants. If $\alpha = \beta = \gamma = 1$, then SSIM index is simplified as:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{3.22}$$

SSIM index satisfies following properties:

- Symmetry: $SSIM(x,y) = SSIM(y,x)$
- Bounded: $SSIM(x,y) \leq 1$
- Unique Maximum: $SSIM(x,y) = 1\ iff\ x = y$

3.4.4. Size of Location Map

Location map is an array of bits that stores ‘1’ or ‘0’ to indicate the location in cover image where overflow/underflow can be occurred during embedding operation. The location map is required for correct extraction and lossless image restoration. The size of location map is measured in bits, without applying any lossless compression.

3.5. Applications of Reversible Watermarking

Reversible watermarking, also termed as reversible data hiding, invertible or lossless watermarking, permits lossless recovery of cover image together with the watermark by authentic receivers. Therefore, the restored image is identical with the original cover or host image. Reversible watermarking can be viewed as a superior version of digital watermarking and has many varied applications such as in medical imaging, forensic industries, secure storage, military imaging etc. Some practical applications of reversible watermarking are discussed below.

3.5.1. Integrity Verification of Digital Data

For integrity verification of digital data at receiver end, the digital watermark to be embedded is a combination of cover image integrity value (calculated using RIPEMD, MD5, SHA etc.) and the payload. After receiving the watermarked data or image, the intended receiver will extract the integrity value and then the image will be restored. The receiver will accept the received image only when the obtained integrity value (from extracted watermark) matches with the calculated integrity value of the restored image. A general reversible watermarking system used for checking integrity (at receiver end) of digital image is presented in Fig. 3.5 [6].

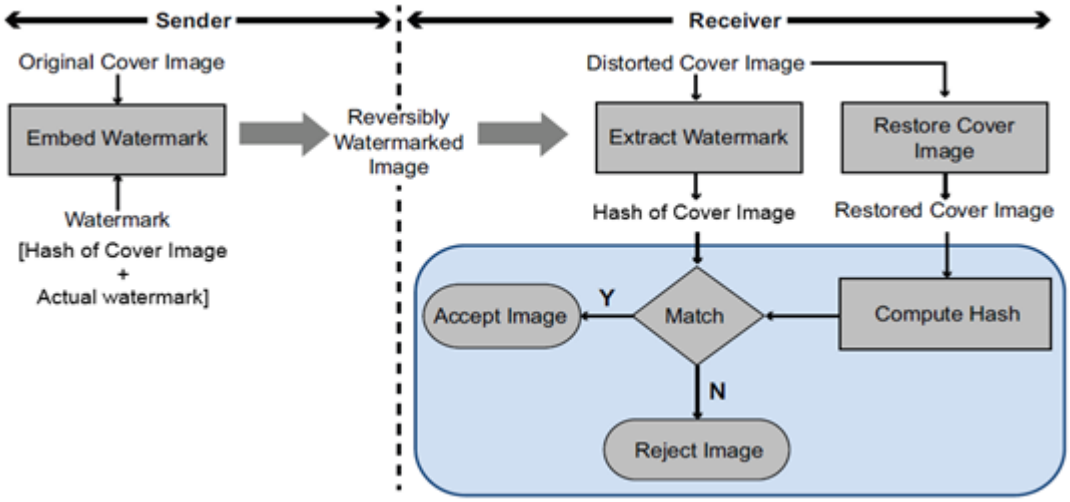


Figure 3.5: Integrity Verification of Digital Multimedia

3.5.2. Reversible Watermarking of Medical Images

Fig 3.6 [30] shows the usage of reversible watermarking system in medical images. In healthcare applications, reversible watermarking scheme is used to embed the patient's credentials like patient record, medical reports etc. into medical image of the patient. The reversibly watermarked medical image is then sent to some other location. At the receiver end, the original medical image (cover image) is recovered after extraction of patient's credentials. Instead, if irreversible digital watermarking was employed, then the original medical image is not exactly recovered and hence reversible watermarking technique is required in case of medical applications.

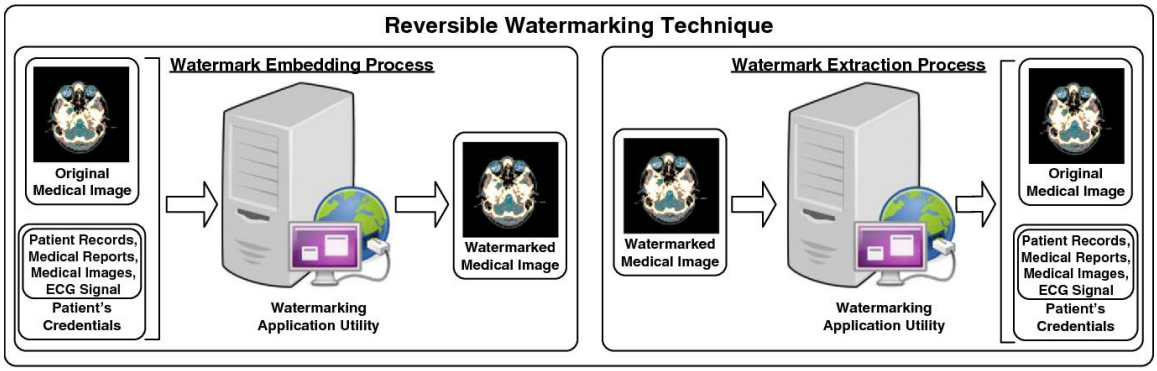


Figure 3.6: Reversible Watermarking of Medical Images

3.5.3. Secure Storage in Cloud

Reversible watermarking can be used for secure storage of digital images in third party clouds as shown in Fig. 3.7 [31]. The owner of the digital image want to securely store the images in third party cloud. To ensure privacy, the user will encrypt the digital images before storing images into cloud. For correct identification and management of the encrypted images, some extra information or labels, such as user's name, timestamp, copyright identification etc. are embedded into encrypted images using reversible watermarking before storing the enciphered images in the cloud. The service provider of the cloud or the authorized receivers are only able to extract the additional embedded labels. The authorized end user will download the encrypted image after removing extra-embedded labels from the cloud and after decryption; the original image can be obtained.

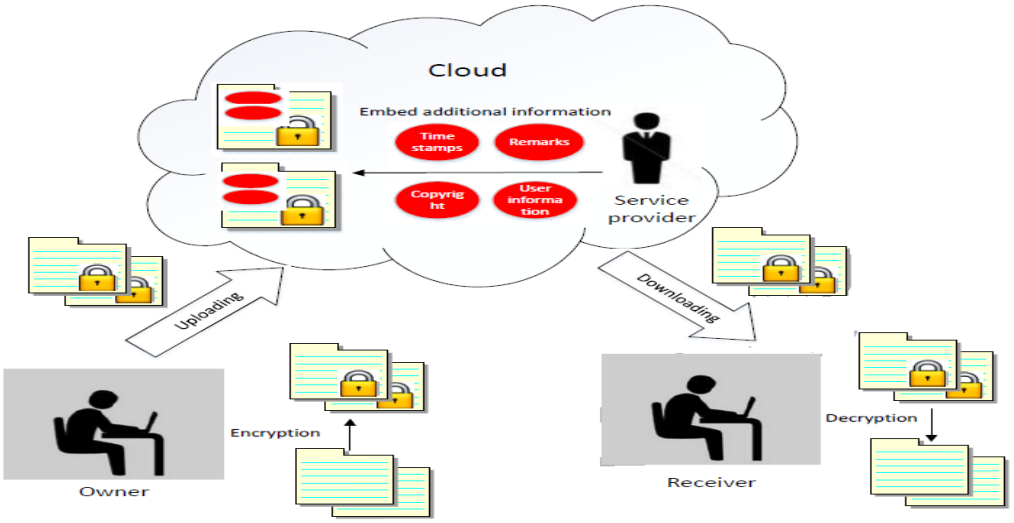


Figure 3.7: Reversible Watermarking in Cloud Storage

CHAPTER 4

METHODOLOGY OF PROPOSED REVERSIBLE WATERMARKING SCHEME

This chapter presents in detail about the proposed three phase reversible watermarking algorithm based on PEE (prediction error expansion) for grayscale digital images. In this chapter, prediction of candidate pixels, phase wise embedding, watermark extraction and image recovery are described in detail. The simulation results of the proposed scheme for some standard and medical test images are mentioned in chapter 6.

4.1. Introduction

The proposed three phase digital image reversible watermarking is based on PEE that utilizes the high correlation present between nearby pixels in digital images. The scheme predicts candidate pixels of three phases using median of equal weighted original context pixels, embedding and extraction is carried out phase wise. Further, showed that only one bit is sufficient for creation of location map that makes embedding process more efficient. In addition, the proposed scheme is extended for localized tamper detection in digital images at the receiver end; details described in chapter 5.

4.2. Prediction Method

In proposed reversible watermarking scheme, the cover image (grayscale) is categorized into three phase candidate pixels namely first, second and third phase candidate pixels and base pixels as introduced in [20]. Base pixels in grayscale image are treated as reference pixels and thus remains intact in the watermarked image while three

phase candidate pixels are utilized for watermark embedding. For each phase, the candidate pixels are predicted with the help of median of the original context pixels, then the prediction error is computed by subtracting the predicted pixel value from corresponding candidate pixel value which are used to embed the watermark. Usually the adjacent pixel intensities in grayscale cover image are highly correlated with each other; therefore, the median of original context pixels provides good prediction and thus producing relatively small magnitude prediction error for most of the candidate pixel. The location of base pixels (indicated by 0's in image) and three phase candidate pixels (indicated by 1, 2 & 3's) in cover grayscale image along with context pixels used for prediction (shown by shaded locations for circled candidate pixels) is shown in Fig. 4.1.

0	3	0	3	0	3	0	3	0	3
2	1	2	1	2	1	2	1	2	1
0	3	0	3	0	3	0	3	0	3
2	1	(2)	1	2	1	2	(1)	2	1
0	3	0	3	0	3	0	3	0	3
2	1	2	1	2	1	2	1	2	1
0	3	0	3	0	3	0	3	0	3
2	1	2	1	2	1	2	1	2	1
0	3	0	3	0	(3)	0	3	0	3
2	1	2	1	2	1	2	1	2	1

Figure 4.1: Base Pixels & Three Phase Candidate Pixels with Prediction Contexts

Let $P(i,j)\ and\ P'(i,j)$ denotes actual and predicted intensity values at $(i,j)^{th}$ candidate pixel location. Then, the three phase candidate pixels of a grayscale cover image are predicted as,

A. First Phase pixels: Location of these pixels represented by 1's in grayscale cover image. To predict the $(i,j)^{th}$ pixel of this phase, the four base pixels around them are utilized as,

$$P'(i,j) = Median[P(i-1,j-1), P(i-1,j+1), P(i+1,j-1), P(i+1,j+1)] \quad (4.1)$$

B. Second Phase Pixels: Location of these pixels indicated by 2's in grayscale cover image. To predict the $(i,j)^{th}$ candidate pixel of 2nd phase, two 1st phase and two base pixels are employed as,

$$P'(i,j) = Median[P(i,j-1), P(i,j+1), P(i-1,j), P(i+1,j)] \quad (4.2)$$

C. Third Phase Pixels: Location of these pixels indicated by 3's in grayscale cover image. To predict the $(i,j)^{th}$ candidate pixel of 3rd phase, two 1st phase and two base pixels are employed as,

$$P'(i,j) = Median[P(i,j-1), P(i,j+1), P(i-1,j), P(i+1,j)] \quad (4.3)$$

For each phase, prediction errors are computed as the difference between candidate pixel and the corresponding predicted value given by equation (4.4) as,

$$e(i,j) = P(i,j) - P'(i,j) \quad (4.4)$$

4.3. Watermark Embedding Algorithm

Reversible embedding of watermark in grayscale cover image carried out phase wise and has following main steps:

a) Locate base and three phase pixels in cover image
b) Prediction of candidate pixels in three phases
c) Computing prediction errors for each candidate pixels
d) Sorting of particular phase prediction errors in ascending order of the variance of original context pixels
e) Embedding of watermark bits into prediction errors

During embedding process, the cover grayscale image is categorized into base pixels and three phase candidate pixels as discussed in section 4.2., base pixels are treated as reference and thus remains intact in watermarked image while candidate pixels are utilized for embedding. The embedding is carried out phase wise i.e. embedding procedure started with first phase candidate pixels, followed by second phase candidate pixels and then in the candidate pixels of third phase.

Before watermark embedding, the prediction errors in a particular phase are sorted in ascending order of the context pixels variance, this will result in better PSNR value when the payload size is small [21]. Thus, embedding started from the candidate pixels with small variance. The computation of context pixels variance for the candidate pixel P given by equation (4.5) as,

$$var(P) = \frac{1}{N}\sum_{n=1}^{N}(P_n - \mu)^2 \quad (4.5)$$

where, N is the total context pixels used for predicting candidate pixel P and μ represents mean value of context pixels. The steps of embedding procedure is given in Algorithm 1 and flowchart for embedding process after finding the parameters is shown in Fig. 4.2.

ALGORITHM-1: EMBEDDING PROCEDURE

```
Input: Cover Image (P), watermark (W)
Output: Watermarked Image (P_wm), location map
[k1, k2] = find_parameters (P, W) /* Find Error Thresholds */
for (Phase m ← 1to 3) do
  for i = 1 to (#Pixels in Phase m) do
    calculate P_i' ← Predicted value of candidate pixel P_i
    compute Prediction error, e_i = P_i − P_i'
    compute variance(P_i) ← context pixels variance for pixel P_i
    array(i, 1) ← e_i
    array(i, 2) ← variance(P_i)
  endfor
  array ← sortrows(array, 2) /*Sort prediction error according to
   ascending order of variance*/
  for l = 1 to size(array) do
    Embed watermark in sorted prediction error & generate location
    map (if required)
  endfor
endfor
RETURN Watermarked Image (P_wm) and location map
```

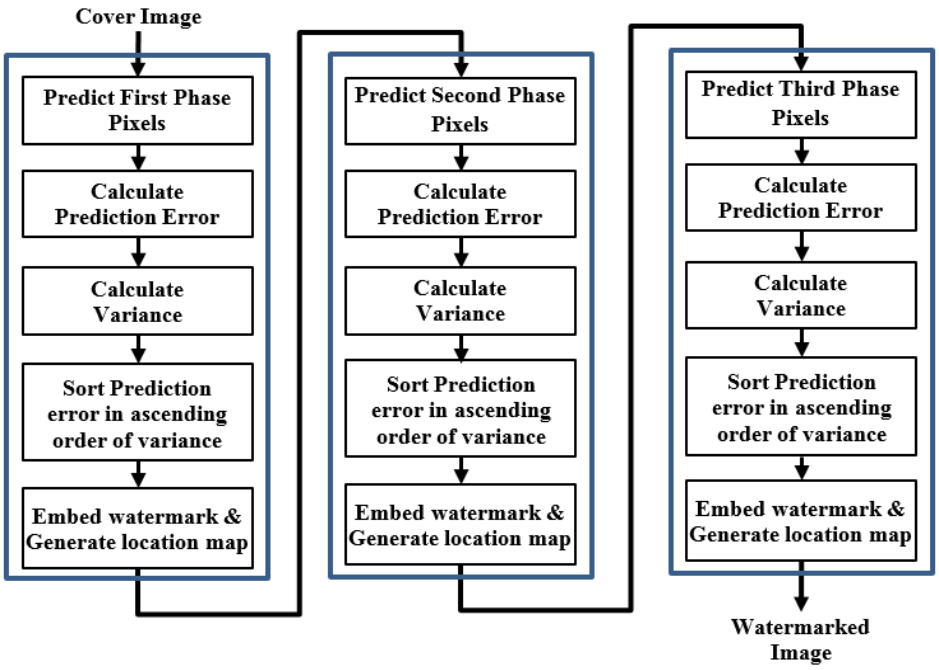


Figure 4.2: Flowchart of the Proposed Embedding Scheme

4.3.1. Embedding of Watermark into Prediction Errors

Prediction error histogram (PEH) i.e. frequency of occurrence of prediction errors are modified to embed the watermark. PEH bins for prediction errors $-k_1 \leq e \leq k_2$ (where, $k_1 and\ k_2$ are error thresholds) are only expanded to embed a watermark bit $b \in \{0,1\}$. For embedding, such prediction errors are multiplied by two and the bit b is added into the LSB. In order to avoid overlapping between the modified prediction errors (after embedding bits) with other prediction errors; prediction errors $e > k_2$ are increased by an amount of $(k_2 + 1)$. Similarly for $e < -k_1$, magnitude k_1 is subtracted from such errors.

Let $\phi(e)$ represents modified prediction error and P_{wm} represents watermarked pixel, then these quantities are given by equation (4.6) and (4.7)

$$\phi(e(i,j)) = \begin{cases} 2e(i,j) + b & if\, e(i,j) \in [-k_1, k_2] \\ e(i,j) + k_2 + 1 & if \quad e(i,j) > k_2 \\ e(i,j) - k_1 & if \quad e(i,j) < -k_1 \end{cases} \tag{4.6}$$

$$P_{wm}(i,j) = \phi(e(i,j)) + P'(i,j) = \phi\big(e(i,j)\big) + P(i,j) - e(i,j) \tag{4.7}$$

For example, for error thresholds $k_1 = k_2 = 2$, Fig. 4.3 shows the transformation of prediction error e into expanded prediction error $\phi(e)$. In this example, prediction errors $-2 \leq e \leq 2$ will be utilized to embed the bits while for $e > 2$, the prediction errors are shifted by constant value 3 and for $e < -2$, the prediction errors are shifted left by constant 2 to avoid overlapping with the bit embedded prediction errors.

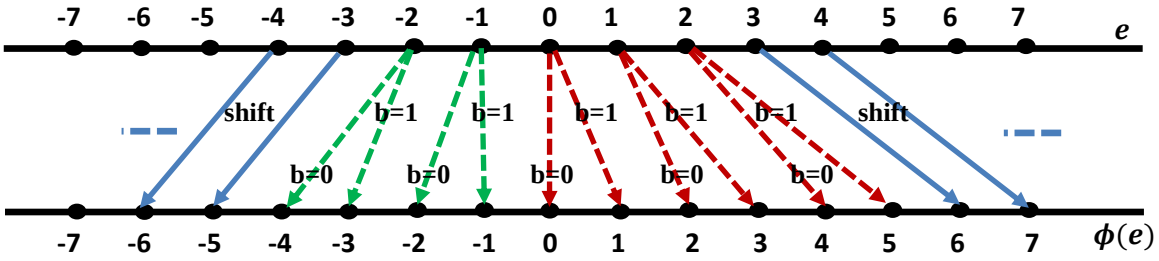
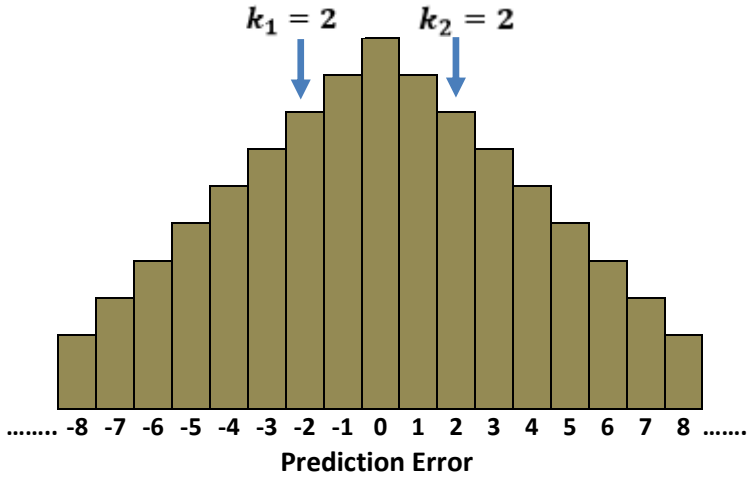
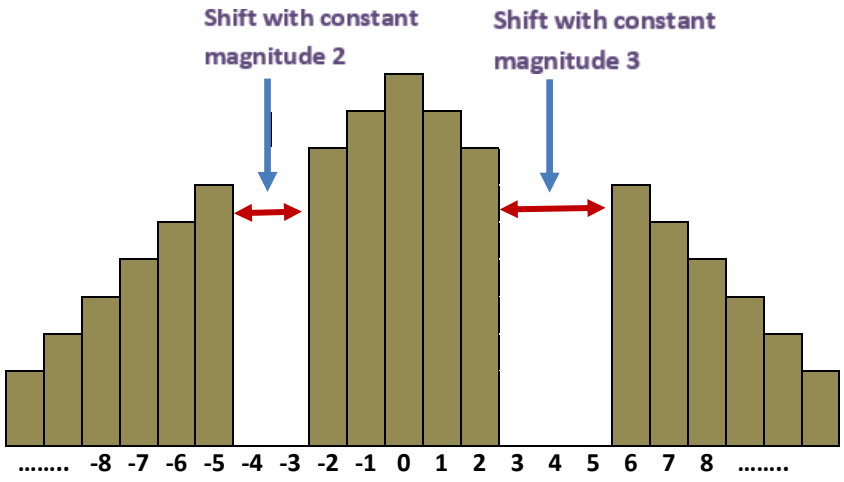


Figure 4.3: Prediction Error Modification for $k_1 = k_2 = 2$

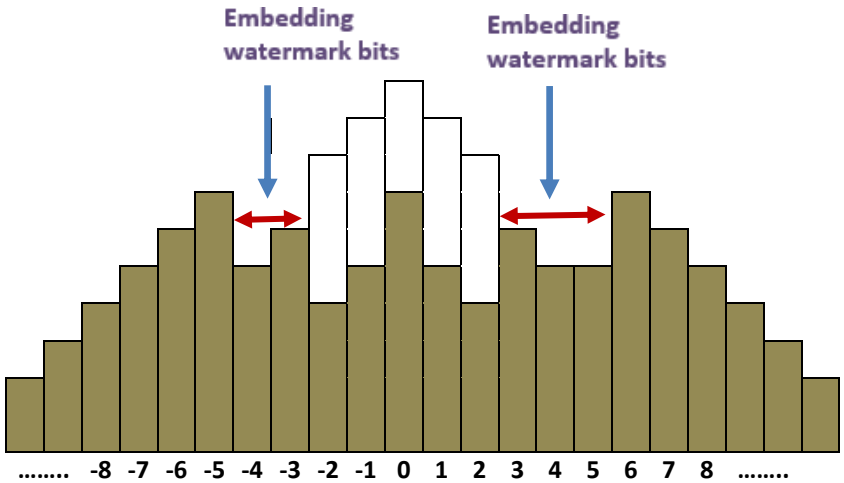
As an example, PEH for an image shown in Fig. 4.4 (a) with error thresholds $k_1 = k_2 = 2$. In this example, bins in the range $[b_{-4}, b_{-3}]$ are emptied by shifting the bins left by k_1 which are equal to or smaller than b_{-3}. Similarly, bins $[b_3, b_5]$ are emptied by shifting the bins right by $(k_2 + 1)$ which are equal to or larger than b_3 as illustrated in Fig. 4.4 (b). Fig. 4.4 (c) shows the embedding of watermark by expanding bins $[b_{-2}, b_2]$.



(a) Prediction Error Histogram (PEH)



(b) PEH after Constant Shift



(c) PEH after Embedding Watermark Bits

Figure 4.4: Example of Prediction Error Histogram Modification for $k_1 = k_2 = 2$

The threshold limits $k_1 and\ k_2$ can be found by exhaustive search mentioned in Algorithm-2. For finding the values of thresholds, the cover image and the actual length of watermark, len is taken. Since some of the watermarked pixels may lead to overflow/underflow, some extra bits have to be considered apart from actual watermark.

ALGORITHM-2: FIND PARAMETERS (k_1 *and* k_2)

```
Input: Cover Image (P), watermark length (len)
Output: Threshold limits (k1 and k2)
for (i ← 1 to (#Phase1 pixels)) do
    compute P' = predict_pixels (P)
    arr1(i) ← P − P'   //1st Phase Prediction error
endfor
for (i ← 1 to (#Phase2 pixels)) do
    compute P' = predict_pixels (P)
    arr2(i) ← P − P'   //2nd Phase Prediction error
endfor
for (i ← 1 to (#Phase3 pixels)) do
    compute P' = predict_pixels (P)
    arr3(i) ← P − P'   //3rd Phase Prediction error
endfor
arr=[arr1,arr2,arr3] /*Concatenate prediction errors*/
k1 = k2 = 0 and i=0
while(1)
    i = i + 1
    if (sum((arr ≥ −k1)||(arr ≤ k2)) ≥ len) then
        break;
    elseif (mod(i,2)!=0) then
        k1 = k1 + 1
    else k2 = k2 + 1
    endif
endwhile
Return k1 and k2
```

4.3.2. Test for Overflow/Underflow

During transformation of pixels from original candidate pixels to watermarked pixels, some of them may produce overflow i.e., $P_{wm} > 255$ or underflow i.e., $P_{wm} < 0$ for $uint\ 8$ bit grayscale image. Such pixels are not transformed and maintained as original values in marked image, the track of such pixel locations are kept in an array called location map. For the case of underflow we have,

$$P_{wm} < 0 \Rightarrow P_{wm} = P' + \phi(e) < 0 \tag{4.8}$$

Since, P' (predicted value) is always greater than or equal to zero i.e. positive, therefore $\phi(e)$ and e are negative. In the second case, i.e., for overflow we have,

$$P_{wm} > 255 \Rightarrow P' + \phi(e) > 255 \quad (4.9)$$

Since, P' is always less than or equal to 255 (for 8 bit pixels), we get always positive $\phi(e)$ and therefore the prediction errors e are also always positive. Hence, from the above observations, conclusion is made that the overflow in the pixels are occurred by positive e while the underflow condition is caused by negative prediction errors.

4.3.3. Location Map Generation

During embedding, location map is also generated (if required) to track over/underflow pixels. Location map is an array of bits that can store '0' or '1' depend on overflow/underflow occurred while embedding the watermark bit in the candidate pixel or occurred while embedding the bit in the transformed pixel obtained after embedding. In other words,

- Location map will store bit '0', if overflow / underflow takes place while embedding the watermark bit in the original candidate pixel.
- Location map will store bit '1', if the embedding of the watermark bit in candidate pixel does not lead to overflow/underflow but occurred if embedding done in watermarked pixel. This entry is utilized to avoid uncertainty during extraction and cover recovery process.

Let the watermarked pixel is denoted by P_{wm}, for creation of the location map, bits '0' and '1' are used as watermark which is embedded in this pixel to test for possible overflow/underflow.

Let $\phi(e) = P_{wm} - P'$ denotes the prediction error for the watermarked pixel. When prediction error $\phi(e) \in [-k_1, k_2]$, it will be modified as $\phi'(e) = 2\phi(e) + b$, where b represents the bit that can be considered '0' or '1'. Then the conditions for overflow and underflow is given by equations (4.10) and (4.11),

$$\text{Underflow Condition} \quad : \quad \phi'(e) + P' < 0 \quad (4.10)$$

$$\text{Overflow Condition} \quad : \quad \phi'(e) + P' > 255 \quad (4.11)$$

Case I: Underflow- In this case $P' + \phi'(e) = P' + 2\phi(e) + b < 0$, bit b can be considered as '0' or '1'. The location map will have an entry '1' if for any of the bit value the underflow will occur. Therefore, rather than checking the condition of underflow with

both bits, it is sufficient to check the condition with bit $b = 0$ i.e. the condition for testing underflow becomes $P' + (2 \times \phi(e)) < 0$.

Case II: Overflow- The condition for overflow is, $P' + \phi'(e) = P' + 2\phi(e) + b > 255$. In the case of overflow, checking overflow for bit $b = 1$ is sufficient, in other words the condition for testing overflow becomes, $P' + 2\phi(e) > 254$.

4.3.4. Flowchart of Watermark Embedding into Prediction Error

Flowchart of complete embedding procedure for expanding the prediction error (either by embedding bits into prediction error or by shifting the prediction errors by constant value), location map generation (if required) and formation of watermarked pixel is shown in Fig. 4.5.

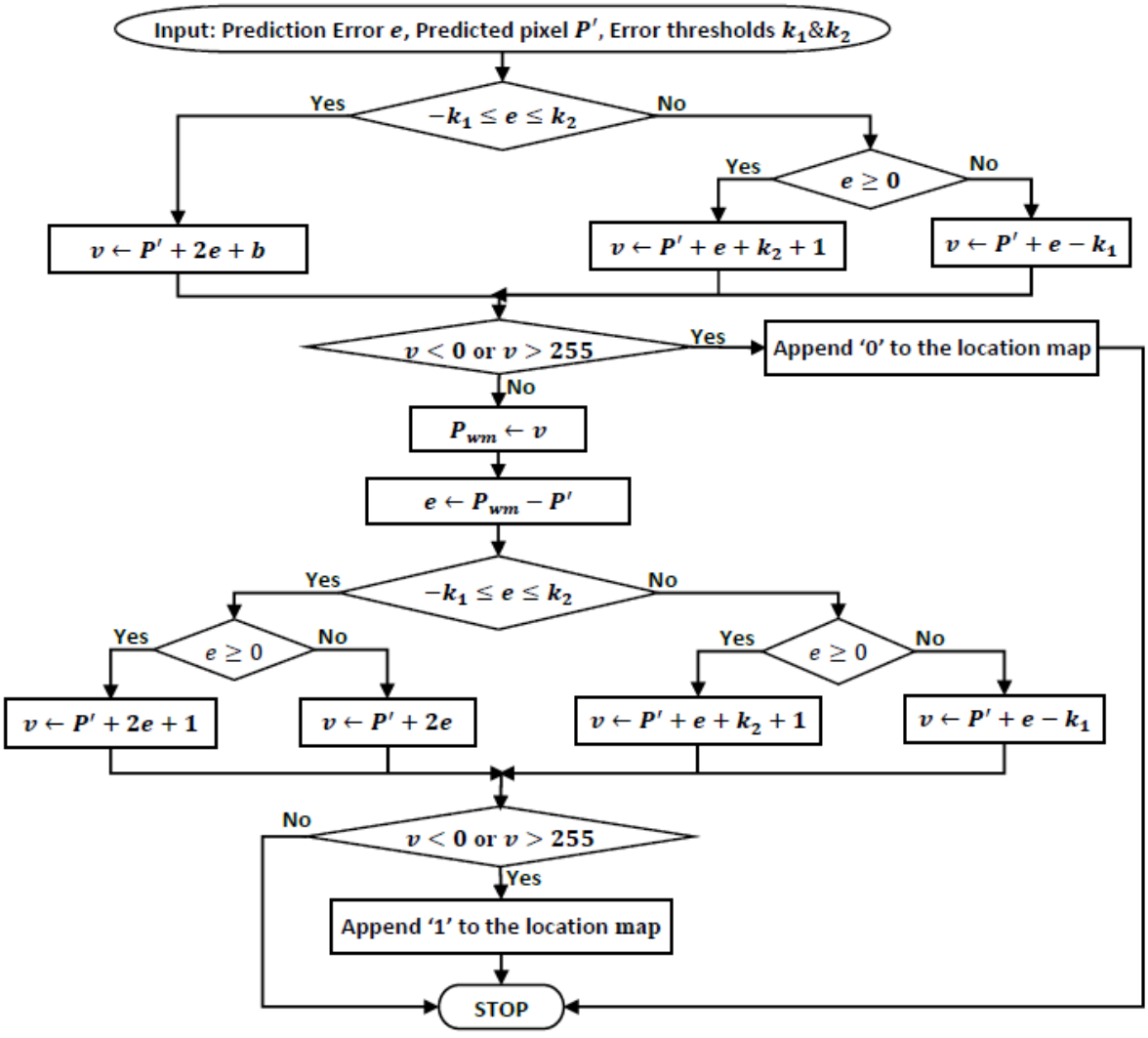


Figure 4.5: Flowchart for Embedding Watermark in Prediction Error

4.4. Watermark Extraction and Recovery of Cover Image

For watermark extraction and recovery of cover image, the watermarked image pixels are categorized into base pixels and three phase candidate pixels similar to embedding process. Then extraction process is carried out phase wise similar to embedding process i.e. after complete processing of first phase pixels, second phase pixels followed by pixels at third phase positions are processed, until the complete watermark and the entire original image pixels are restored.

The extraction process is started with finding the modified or expanded prediction error $\left(\phi(e)\right)$. The expanded prediction error $(\phi(e))$ is computed using two variables viz., watermarked pixel (P_{wm}) and the corresponding predicted pixel (P') at that position. Since $\phi(e)$ is computed successively for three phases, the prediction context of the watermarked candidate pixels for all three phases remains identical with that of cover image and therefore, the predicted values for cover and watermarked image are identical i.e., P' and P'_{wm} are identical. Therefore, the modified prediction error is given by equation (4.12).

$$\phi(e) = P_{wm} - P'_{wm} = P_{wm} - P' \tag{4.12}$$

After computing $\phi(e)$ for candidate pixels in watermarked image, the modified prediction errors of a particular phase are sorted in ascending order of the variance of original neighbouring context pixels. Note that the variance of context pixels for marked image candidate pixels is identical with that of the cover image, as the prediction context remains same. Then, watermark bit (if embedded) is extracted followed by recovery of actual prediction error (e) and then, the original pixels of cover image are restored. The steps for watermark extraction and complete recovery of image from the given watermarked image, location map and threshold values are mentioned in Algorithm 3.

ALGORITHM-3: WATERMARK EXTRACTION & IMAGE RECOVERY

```
Input: Watermarked Image (P_wm), location map (loc), k_1 and k_2
Output: Watermark and cover image (P)
  /* Extraction and Recovery from First Phase */
  for (i ← 1 to (#Phase1 pixels)) do
   compute P_i' = predict_pixels (P_wm_i)  //using Base pixels
   array(i,1) ← φ(e_i) = P_wm_i − P_i'  //Mod. Prediction Error
   array(i,2) ← var(P_wm_i) //context pixels variance for P_wm_i
  endfor
  array = sortrows(array,2) //Sort array rows in ascending order of
  variance
  for m = 1 to length(array) do
    Extract watermark bit (if present) and restore pixels with the
    help of location map
  endfor

   /* Extraction and Recovery from Second Phase */
  for (i ← 1 to (#Phase2 pixels)) do
   compute P_i' = predict_pixels (P_wm_i)  //using Base and recovered
   First Phase pixels
   array(i,1) ← φ(e_i) = P_wm_i − P_i'  // Mod. Prediction Error
   array(i,2) ← var(P_wm_i) //context pixels variance for P_wm_i
  endfor
  array = sortrows(array,2) //Sort array rows in ascending order of
  variance
  for m = 1 to length(array) do
   Extract watermark bit (if present) and restore pixels with the
   help of location map
  endfor

   /* Extraction and Recovery from Third Phase */
  for (i ← 1 to (#Phase3 pixels)) do
   compute P_i' = predict_pixels (P_wm_i)  //using Base and recovered
   First Phase pixels
   array(i,1) ← φ(e_i) = P_wm_i − P_i'  // Mod. Prediction Error
   array(i,2) ← var(P_wm_i) //context pixels variance for P_wm_i
  endfor
  array = sortrows(array,2) //Sort array rows in ascending order of
  variance
  for m = 1 to length(array) do
    Extract watermark bit (if present) and restore pixels with the
    help of location map
  endfor
```

The watermarked pixels with $\phi(e) \in [-2k_1;\ 2k_2 + 1]$ are utilized for watermark bit extraction from LSB of $\phi(e)$ and subsequently the actual error e is computed, while for other cases only the actual error e is computed. For $\phi(e) > (2k_2 + 1)$, the actual prediction error is found by subtracting $(k_2 + 1)$ from $\phi(e)$ since this case

represents that e is always enlarged by $(k_2 + 1)$. Similarly, when $\phi(e) < -2k_1$, original prediction error is found by adding k_1 to $\phi(e)$. Therefore, the embedded bit (b) and actual prediction error (e) can be found from $\phi(e)$ using equations (4.13) and (4.14).

$$b = \phi(e)(mod\ 2)\ \ if\ -2k_1 \leq \phi(e) \leq (2k_2 + 1) \tag{4.13}$$

$$e = \begin{cases} \dfrac{\phi(e) - b}{2} & if \phi(e) \in [-2k_1, 2k_2 + 1] \\ \phi(e) - (k_2 + 1) & if \quad \phi(e) > 2k_2 + 1 \\ \phi(e) + k_1 & if \quad \phi(e) < -2k_1 \end{cases} \tag{4.14}$$

After computing actual prediction errors (e), for each candidate watermarked pixel, cover image pixels are obtained by using equation (4.15).

$$P = P' + e \tag{4.15}$$

4.4.1. Utilization of Location Map

For extraction and recovery of original pixel from each candidate pixel of watermarked image, first the dummy watermarked pixel is generated using the procedure discussed in section 4.3, then this pixel is checked for possible overflow or underflow. For generation of this pixel, only single bit is required to embed in each candidate-watermarked pixel as described in section 4.3.3. If any of the candidate pixel in watermarked image led to overflow/underflow during this process, then the bit stored in *'Location Map'* is accessed. If the location map has a bit '0', then the candidate pixel is not processed. If the location map has a bit '1', then the candidate pixel is processed to extract the watermark bit (if embedded) and then actual pixel is recovered.

4.4.2. Flowchart of Watermark Extraction and Image Recovery

Flowchart for extracting the watermarked bit from modified prediction errors and restoration of original pixel with the help of location map shown in Fig. 4.6.

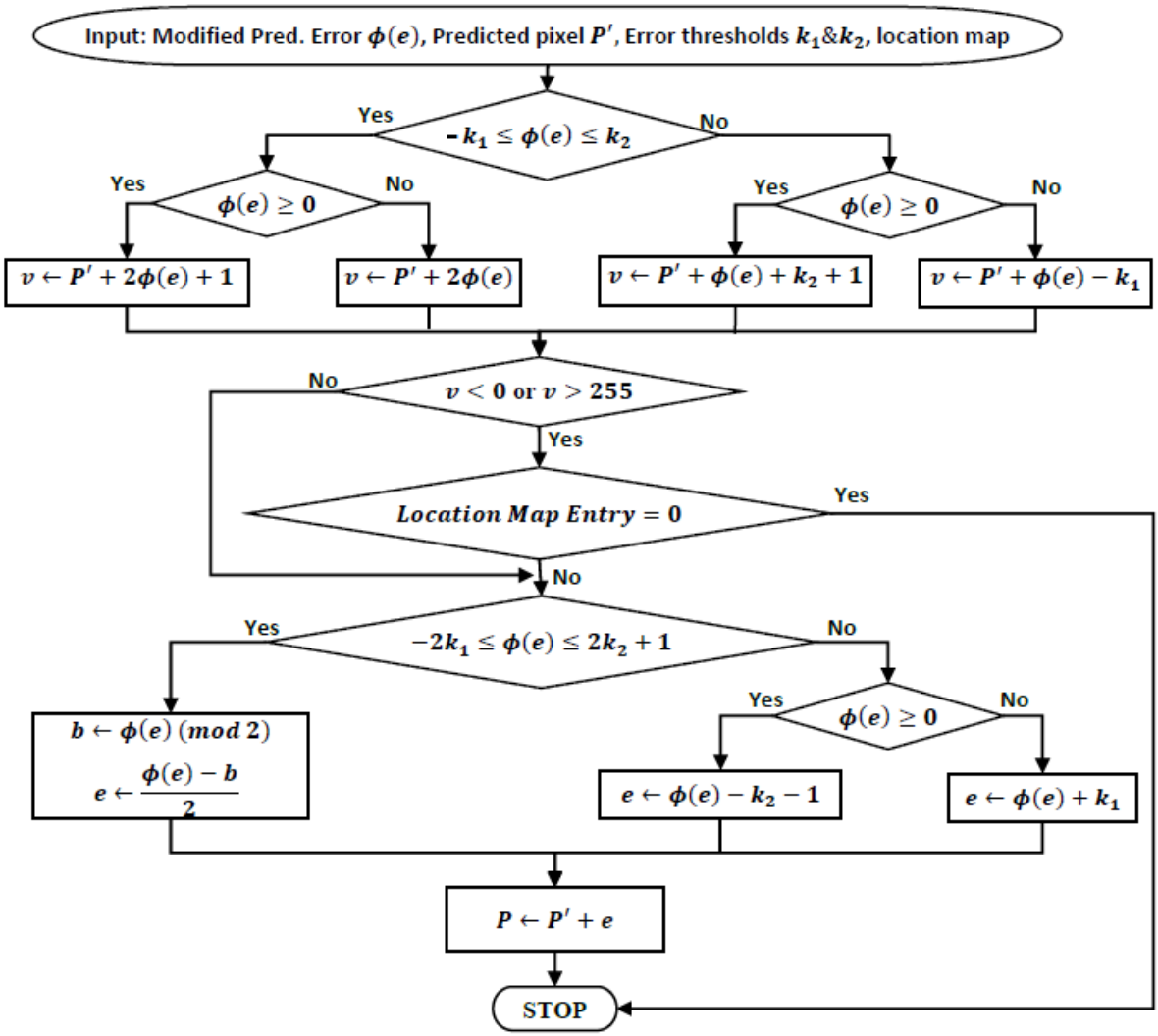


Figure 4.6: Flowchart for Watermark Extraction and Image Restoration

4.5. Side Information for Blind Extraction

For perfect watermark extraction and complete retrieval of cover image, the receiver requires side information (i.e. threshold values, location map and payload size). The proposed reversible scheme can be converted into blind reversible scheme by inserting the side information in addition to the actual payload in the cover image to produce final watermarked image. This enables the receiver to obtain the watermark and retrieve the cover image without requiring any additional external information.

For experimentation purpose, number of bits assigned for different fields of 'side information' is depicted in Fig. 4.7. The threshold values k_1& k_2 are considered in the range [0,10], hence 4 bits are required for each threshold.

Error Threshold, k_1	Error Threshold, k_2	Watermark length	Length of location map	Location map
4 bits	4 bits	20 bits	10 bits	variable

Figure 4.7: Side Information for Extraction

The side information bits are inserted in the image by overwriting the LSBs (least significant bits) of first phase pixels (started from candidate pixels with maximum context pixel variance). Let n_s represents the total size of side information, then n_s number of pixels are used for insertion of side information. For lossless restoration of cover image, the actual LSBs of first phase pixels (utilized for insertion of side information) are saved in one dimensional array and combined with actual watermark to form the effective payload.

During the process of extraction and recovery of host image, first the side information (i.e. threshold values, location map and payload size) are obtained from the LSBs of first phase pixels (started from the candidate pixels with maximum variance), then watermark is extracted and pixels are recovered from the marked image. Finally, from the initially n_s number of extracted watermark bits, the first phase pixels (utilized for side information) are bring to actual values.

CHAPTER 5

LOCALIZED TAMPER DETECTION USING PROPOSED REVERSIBLE WATERMARKING SCHEME

This chapter presents the application of the proposed reversible watermarking scheme for tamper localization in digital images. Localization of tampered areas allows the recipient to reject only selective areas in case of integrity/authentication failure. The algorithm for embedding and detection of tampered areas were implemented in MATLAB to evaluate the effectiveness, implementation results are shown in chapter 6.

5.1. Introduction

The prime requirement for industries dealing with sensitive data is to ensure the authentication and integrity of the received data. Tampering in digital image during transmission can occur due to different types of unintentional and intentional attacks that may affect complete or part of the image. Tamper detection and authentication in digital images can be achieved through digital signature and digital watermarking [32]. Digital signature based methods provides tamper detection but does not localize the tampered regions in received image. Fragile digital watermarking provides tamper detection by embedding checksum data in an image [33] [34] but it modifies the host image permanently. To overcome this, reversible digital watermarking which is in general fragile in nature and permits lossless recovery of image can be employed for tamper detection of sensitive data. General framework for verification of integrity of complete received image employing reversible scheme is shown in Fig. 5.1. [6]

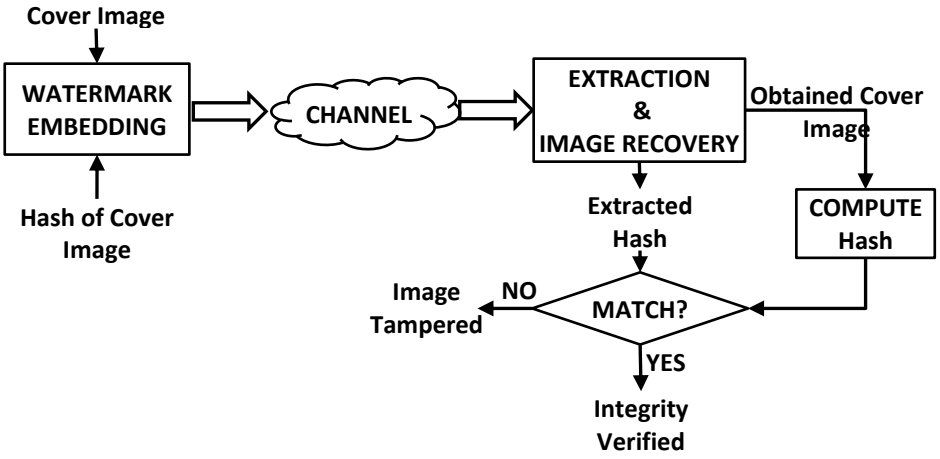


Figure 5.1: Integrity Verification using Reversible Watermarking

Depending on the channel condition and attack scenario, different integrity check algorithms such as additive checksum, CRC, hash or MAC function can be used to generate a watermark. If integrity of the whole cover image is used as watermark, then even a single bit corruption in image produces authentication/integrity failure that results in rejecting the received image and the complete image will be retransmitted.

Localization of tampered areas is achieved by dividing the image into small regions or blocks and embedding the integrity values into the respective regions, it permits selective rejection of tampered areas in case of integrity failure. Naskar et. al. [35] introduced generalized process for tamper detection and localization in digital images using reversible watermarking. They discuss a lower bound on minimum block size based on length of checksum and side information, and propose merging of four adjacent blocks when the embedding capacity is not sufficient. As the method does not consider block location while computation of checksum, it does not provide protection against block copy and move type of attack.

In this chapter, localized tamper detection in digital images using three-phase reversible scheme is presented. In this scheme, the integrity value for each block is computed taking into account within block pixels and block number (to prevent copy-move attack) and allows combining of current and next blocks dynamically when capacity of a current block is not sufficient. The advantages of localization of tampered areas in the received image are:

- In case the corruption occurs in some regions of non-interest (RONI) then the received image will be accepted without request of retransmission.

- In other scenario, where some regions of interest (ROI) are tampered then only that regions will be retransmitted, resulting in saving of the channel bandwidth.

5.2. Embedding Process

The watermark embedding for tamper localization using the proposed reversible watermarking scheme (described in chapter 4) is carried out by partitioning the cover image into small sized blocks, and then integrity value of each individual block is calculated (taking into account the block pixels along with block number), and embedded into corresponding block considering integrity value as a watermark. General embedding process for localized tamper detection is presented in Fig. 5.2.

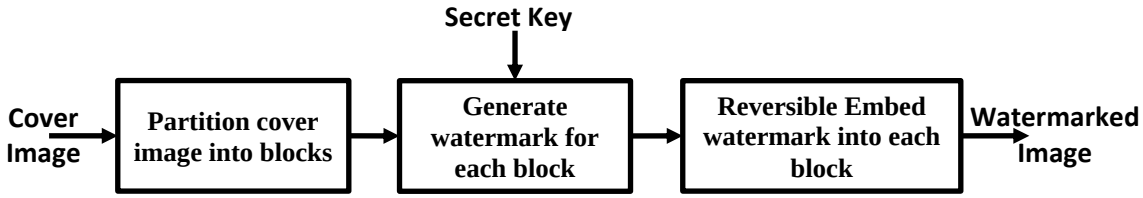


Figure 5.2: Embedding for Tamper Localization

For the purpose of localized tamper detection (at the receiver end), embedding is carried in following steps:

Step 1: Partition of Cover Image into Blocks- The cover image is partitioned into equal sized blocks (non-overlapping), each of size $n_1 \times n_2$ pixels, where $n_1 \& n_2$ represents number of rows and columns in the block. The block parameters are selected in such a way that,

- Blocks should be of same size i.e., uniform throughout the image
- Total number of blocks must include the complete cover image
- Block size must be capable of embedding the integrity value generated in step 2

Step 2: Generation of Watermark- For each block of size $n_1 \times n_2$ pixels, the watermark is the integrity value (say H) is calculated using suitable algorithm depend on the embedding capacity, channel condition and attack scenario considering each pixel value within the block along with block number (say B) i.e. $H = hash\ (B||Block\ B\ pixels)$.

$$Watermark\ =\ H\ ||\ LSB's\ of\ Phase1\ Pixels\ (used\ for\ side\ information) \qquad (5.1)$$

Step 3: Reversible Watermark Embedding- Within each block, the watermark (integrity value) generated in step 2 is embedded using blind reversible watermarking embedding

algorithm described in chapter 4. Finally, the side information i.e., error thresholds k_1, k_2 and location map (required for extraction process) are also inserted in the LSB's of 1st phase candidate pixels started from the beginning for each respective block.

During embedding of the selected integrity value in blocks, it may possible that for some blocks the embedding capacity (EC) is not sufficient. In order to handle this situation, the current block and the next block is combined together and the integrity value of the combined block is embedded. In order to facilitate the receiver, an indicator '0' or '1' for normal /combined block is inserted at the beginning of the side information. Embedding process for localized tamper detection is given in Algorithm 4.

ALGORITHM-4: EMBEDDING PROCESS FOR LOCALIZED TAMPER DETECTION

1. **Input:**
   Cover Image : P
   Block Size : n_1 *and* n_2
   Integrity Algorithm : Hash //used hash function
   Secret Key : K (if HMAC is used)
2. **Output:**
   Watermarked Image : P_{wm}
3. Partition cover into Blocks of size $n_1 \times n_2$
4. $B \leftarrow 1$ //Block Number
5. **While** $(B \leq (\#Blocks))$ **do**
6. $Indicator \leftarrow 0$ //Bit used to indicate normal/combined Block
7. **Compute** $H \leftarrow Hash\,(B||Block\ B\ Pixels, K)$
8. $Watermark \leftarrow H \cup LSBs\ of\ 1^{st} Phase\ Pixels\ (used\ for\ side\ information)$
9. $[k_1, k_2, EC] = find\_parameters(Block\ B, Watermark)$
10. **if** $(EC(Block\ B) \geq length(Watermark))$ **then**
11. **Embed** watermark in block B and Generate Location Map
12. **Insert** side information with 1^{st} bit '0' as indicator
13. $B \leftarrow B + 1$ //Increment B by 1
14. **else**
15. $Indicator \leftarrow 1$ //Bit used to indicate combined Block
16. $B_{comb} \leftarrow Block\ B\ ||\ Block\ (B+1)$ //Combine current and next block
17. **Compute** $H \leftarrow Hash\,(B||B_{comb}\ Pixels, K)$
18. $Watermark \leftarrow H \cup LSBs\ of\ 1^{st} Phase\ Pixels (used\ for\ side\ information)$
19. $[k_1, k_2, EC] = find\_parameters(B_{comb}, Watermark)$
20. **Embed** Watermark in block B_{comb} and Generate Location Map
21. **Insert** side information with 1^{st} bit '1' as indicator
22. $B \leftarrow B + 2$ //Increment B by 2
23. **endif**
24. **end while**
25. $P_{wm} \leftarrow Combine\ all\ watermarked\ blocks\ in\ their\ respective\ location$
26. **RETURN** P_{wm}

5.3. Localized Tamper Detection Process

General framework for watermark extraction, image recovery and localized tamper detection process is presented in Fig. 5.3.

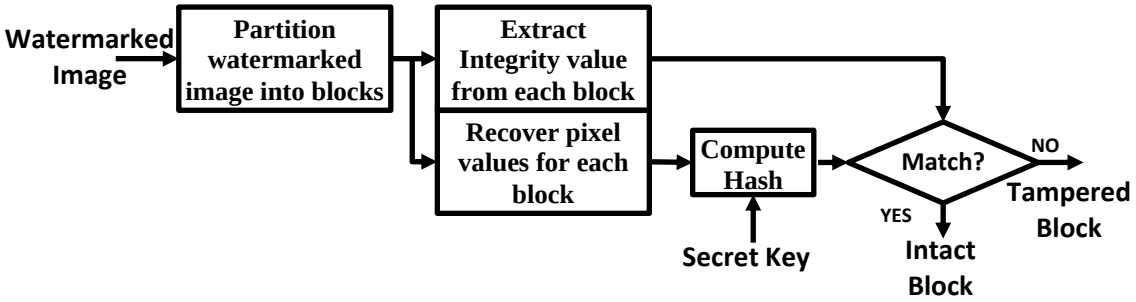


Figure 5.3: Localized Tamper Detection Process

Step 1: Partition of Watermarked Image into Blocks- The received image is partitioned into $n_1 \times n_2$ sized blocks (non-overlapping). Block parameters n_1 and n_2 should be same with the parameters used during the watermark embedding process.

Step 2: Extract Block Parameters- From each block, first the indicator bit is extracted (to know whether the block is normal or combined block) followed by extracting the parameters (k_1 & k_2) and location map from first phase LSBs (utilized for insertion of side information). This information is used for extracting watermark bits and restoring the pixels from each block.

Step 3: Watermark Extraction and Image Recovery- Each block pixels are categorized into base pixels and three phase candidate pixels as described in chapter 4. The watermark extraction and image recovery is carried out phase wise as done during embedding i.e., extraction and recovery started with first phase pixels and then subsequently from second phase and third phase candidate pixels until complete integrity value is obtained. Details of watermark extraction and original cover pixel are described in chapter 4.

Step 4: Tamper Detection- For each block, after extracting the hash value and restoration of pixel values, the hash of the restored pixels along with block number is computed using the same hash algorithm and secret key (in case of HMAC) as used during the watermark generation process. If the computed hash matches with the extracted hash value, then that block is declared intact otherwise the block is tampered.

Steps for localized tamper detection along with pixel recovery from each block of watermarked image is mentioned in Algorithm 5.

ALGORITHM-5: LOCALIZED TAMPER DETECTION PROCESS

```
1. Input:
     Watermarked Image   : P_wm
     Block Size          : n_1 and n_2
     Integrity Algorithm: Hash //used hash function
     Secret Key          : K (if HMAC is used)
2. Output:
     Tampered Blocks     : B_T
     Restored Image      : P_recov

3.  Partition watermarked image into Blocks of size n_1 × n_2
4.  i← 1 and B ← 1 //Block Number
5.  While (B ≤ (#Blocks)) do
6.     Extract Indicator bit from LSB of 1st phase pixels
7.     if (Indicator = 0) then
8.        Extract side information //k_1, k_2 and location map
9.        Extract Hash value (H) and restore pixels of block B
10.       Compute H_comp ← Hash (B||Restored Block B, K)
11.       if (H = H_comp) then
12.          Authenticate and accept block B
13.       else  Reject block B  //Block B is Tampered
14.             B_T(i++) ← B
15.       endif
16.       B ← B + 1
17.   endif
18.   if (Indicator = 1) then
19.      B_comb ← Block B ||Block(B + 1)  //Combine current and next block
20.      Extract side information //k_1, k_2 and location map
21.      Extract Hash value (H) and restore pixels of block B_comb
22.      Compute H_comp ← Hash (B||Restored Block B_comb, K)
23.      if (H = H_comp) then
24.         Authenticate and accept block B_comb
25.      else  Reject block B_comb  //Blocks B & (B + 1) are Tampered
26.            B_T(i++) ← B
27.            B_T(i++) ← B+1
28.      endif
29.      B ← B + 2
30.   endif
31. end while
32. P_recov ← Combine all restored blocks in their respective location
33. RETURN P_recov and B_T
```

5.4. Selection of Integrity Check Algorithm

The integrity value for each block of an image is generated using algorithm based on various types of corruption and attack scenario and used as a watermark. Framework for selection of integrity check algorithm is as follows:

- **Protection Against Accidental Corruption-** For detecting random and unintentional corruption during transmission of digital image from sender to remote end, one can use simple checksum algorithms such as additive, Adler checksum, cyclic redundancy check (CRC) etc. to generate integrity value for the blocks. Since, size of the checksum for these algorithms are relatively small, therefore small size blocks can be used which results in better localization of tampered areas in the received image.

- **Protection Against Intentional Corruption-** To detect the intentional corruption i.e., if someone actively and intelligently modifying the message then to protect against this sort of attack, a cryptographic hash function (like hash computed using RIPEMD, MD5, SHA family etc.) can be used to generate hash value. A cryptographic hash function accepts an arbitrary length message and generates a fixed length output with following properties:

  - ***Pre-Image Resistance:*** For any given value of hash (say, H), it is difficult to find any input data M such that $H = hash(M)$. This is also referred as one-way function.
  - ***Second Pre-Image Resistance:*** For a given message M_1, it is computationally difficult to discover another message M_2 $(M_1 \neq M_2)$ such that $hash(M_1) = hash(M_2)$.
  - ***Collision Resistance:*** It is infeasible to find any message pair (M_1, M_2) such that $hash(M_1) = hash(M_2)$, this property is also referred as strict collision resistance.
  - ***Avalanche:*** Even one bit flip in input changes nearly 50% bits in output.

- **Protection against Malicious Corruption-** In case of malicious corruption or forgery, the attacker will be able to modify both the block pixel values and the embedded integrity value (assuming known embedding procedure). Thus, if simple checksum or cryptographic hash function is used then the forged block seems to be non-tampered. Hence, for protection against this type of attack, message authentication code (MAC) e.g. HMAC (also referred as keyed hash function) must be used, since it involves a secret cryptographic key (shared between transmitter and receiver) for computation of hash value. HMAC provides data integrity as well as the authentication of the source.

Secret key used in MAC can be shared through secure channel between transmitter and receiver. When secure channel is not available then one can use cryptographic protocols (Diffie Hellman, RSA etc.) for sharing of the secret key. The MAC of the data using the HMAC function is computed using equation (5.2) and the steps for HMAC computation provided in Algorithm-6 [36]:

$$MAC = HMAC(K, data) \tag{5.2}$$
$$= Hash((K_0 \oplus opad) || Hash((K_0 \oplus ipad) || data))$$

where, K_0 represents processed secret key K, $ipad$ is 0x36 replicated B_H (block size in $Hash\ function$) times and $opad$ is 0x5C replicated B_H times.

ALGORITHM-6: HMAC CALCULATION

```
1.  Input:
      key(K)                //array containing key bytes
      data                  //whose hash is to be calculated
      hash function(Hash)   //used hash function like MD5,SHA etc.
      outsize               //length of hash function output in bytes
      blocksize(B_H)        //size of block used in the hash function
2.  Output:
      hmac                  //HMAC of the given data with key K

3.    if (length(K) > B_H) then
4.       K_0 ← Hash(K)||00 ... 00  //Pad Hash(K) with 0's to make B_H length
5.    if (length(K) < B_H) then
6.       K_0 ← K||00 ... 00        //Pad K with 0's to make it B_H length
7.    ipad ← byte 0x36 repeated B_H times
8.    opad ← byte 0x5c repeated B_H times
9.    Return hmac = Hash((K_0 ⊕ opad) || Hash((K_0 ⊕ ipad) || data))
```

Unlike checksum or cryptographic hash function, the HMAC value can be generated and verified only by one which have correct knowledge of the secret key and hence prevent against forgery or malicious corruption in the watermarked data.

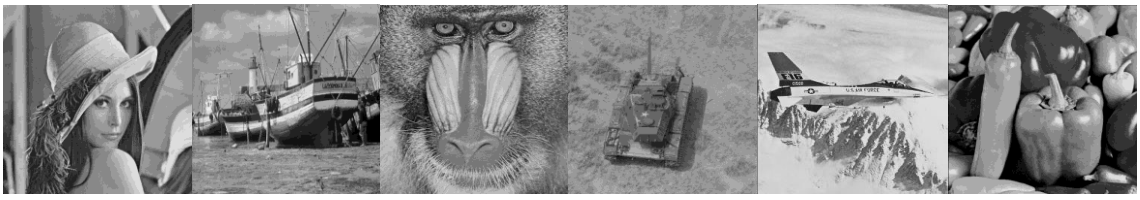
CHAPTER 6

EXPERIMENTAL RESULTS & ANALYSIS

The proposed reversible watermarking scheme for digital images described in chapter 4 was simulated in MATLAB 2013a for performance evaluation and comparison with some state-of-the-art schemes. The performance and effectiveness of the algorithm was tested on some standard and medical test images for maximum embedding capacity (without considering side information), PSNR, SSIM and location map size. This chapter also presents the results for localized tamper detection using the proposed reversible watermarking scheme described in chapter 5.

6.1. Results and Analysis for Proposed Reversible Watermarking Scheme

Sample images used for analyzing the performance and effectiveness of the proposed reversible watermarking scheme is shown in Fig. 6.1 [37], [38]. The integrity value (calculated using SHA-256 algorithm) for the images shown in Fig. 6.1 is mentioned in Table below. In addition, for experimentation purpose, random bits (generated using *randi([0,1],1,length)* command in MATLAB) were taken as watermark for assessing the performance of the scheme. Fig. 6.2 shows watermarked images with embedding capacity, $EC = 10^5$ bits.

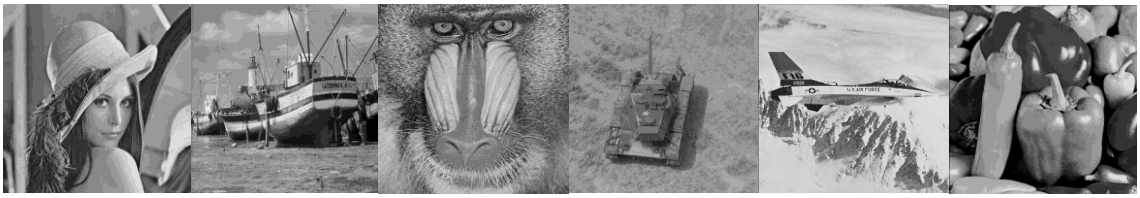


(a) Lena $(\mathbf{512 \times 512})$ **(b) Boat** $(\mathbf{512 \times 512})$ **(c) Baboon** $(\mathbf{512 \times 512})$ **(d) Tank** $(\mathbf{512 \times 512})$ **(c) Airplane** $(\mathbf{512 \times 512})$ **(e) Peppers** $(\mathbf{512 \times 512})$

Figure 6.1: Standard Test Images Used for Analysis

Table III: SHA-256 Values of Test Images

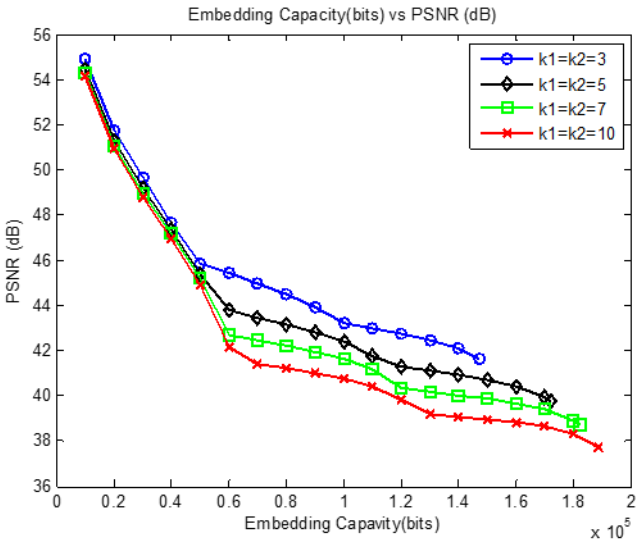
Test Image	SHA-256 value (in hex)
Lena	A4EEF6C3296B1234AB25F1842D99A76B3C7D493A3D781430F7BE6491EBA5A648
Boat	B292548C463580074F3032FDECF2C1873114B797D0827A1E52E9EF37C99989F7
Baboon	B67DDD02C2D2D7CF90E1F4753C9B702655AB6BBD1DD52E5699848083DB5AD216
Tank	8986D7B2ADB6DAEC35D47EF79D5B4711230EFCCDF752B90891A00F195A9E96CF
Airplane	21F27294A0778DAB7CF2D384E7B1868C5E84058A871E43F250DCA298CC75087A
Peppers	F07EBDCC34AB70D15CFD410A77931D9B00C04761D5C41861B5622FF44AB07F1F



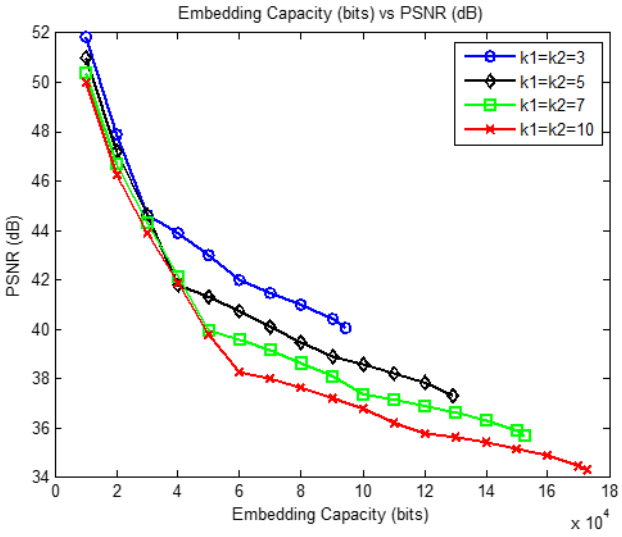
(a) Lena ***PSNR=45.221 SSIM=0.9861*** | **(b) Boat** ***PSNR=39.602 SSIM=0.9696*** | **(c) Baboon** ***PSNR=33.312 SSIM=0.9590*** | **(d) Tank** ***PSNR=39.849 SSIM=0.9674*** | **(c) Airplane** ***PSNR=46.262 SSIM=0.9889*** | **(e) Peppers** ***PSNR=41.898 SSIM= 0.9742***

Figure 6.2: Watermarked Images for $EC = 10^5$ bits

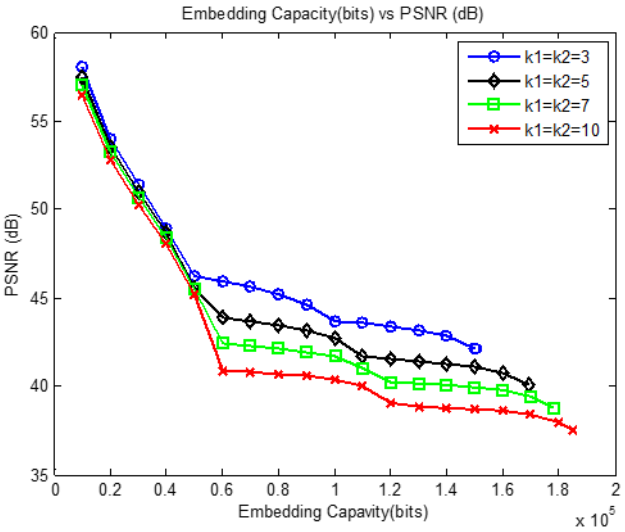
The variation in PSNR with varying size of watermark (in bits) for standard test images with fixed values of error thresholds without considering the overhead/side information is shown in Fig 6.3. These plots shows that the EC (embedding capacity) of the scheme is high with low distortion. As the value of error thresholds increases, EC will increase but at the cost of degraded PSNR as candidate pixels with larger prediction error will also get modified to embed the watermark which results in more degradation in the image.



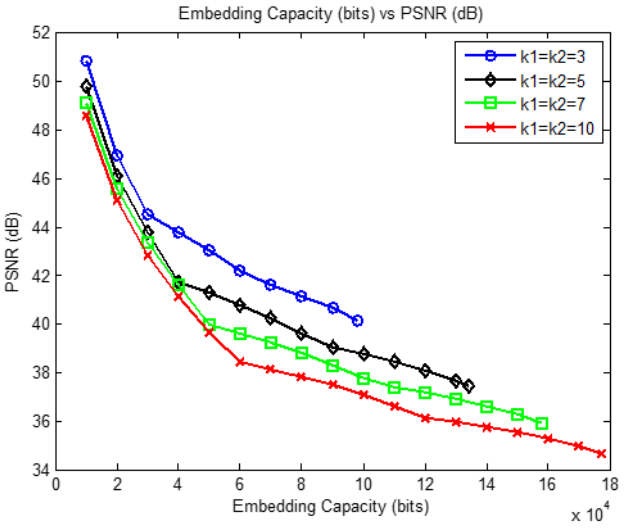
(a) Lena 512 × 512



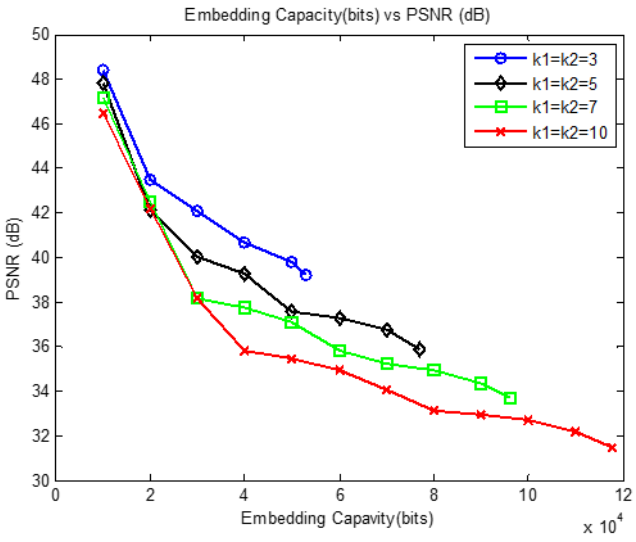
(b) Boat 512 × 512



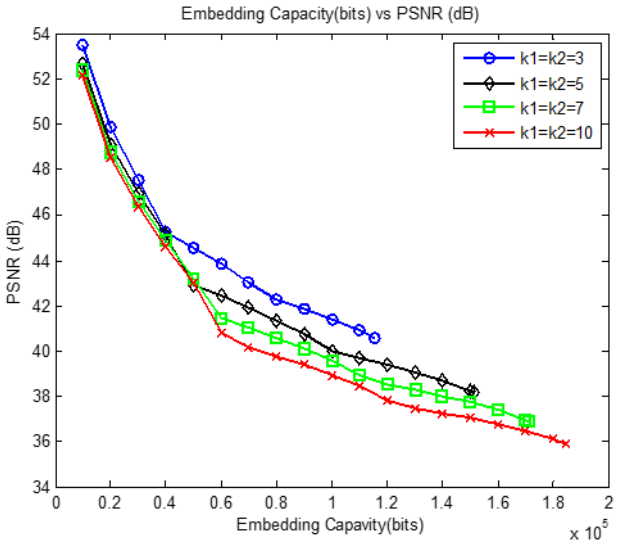
(c) Airplane 512 × 512



(d) Tank 512 × 512



(e) Baboon 512 × 512



(f) Peppers 512 × 512

Figure 6.3: Embedding Capacity vs. PSNR for Fixed Thresholds

Tables below show that analysis of the proposed reversible algorithm with respect to maximum EC (embedding capacity), PSNR and size of location map with fixed error thresholds $k_1, k_2 \in [0,10]$ for standard test images. Results indicate that the scheme has high EC with good PSNR. Also, location map size is very small for most of the cases which makes the smaller overhead information.

Table IV: Results for Lena (512 x 512) and Boat (512 x 512) Grayscale Image

		Lena (512×512)			**Boat** (512×512)		
Thresholds		**Max. Embedding Capacity (bits)**	**PSNR (dB)**	**Location Map Size (bits)**	**Max. Embedding Capacity (bits)**	**PSNR (dB)**	**Location Map Size (bits)**
k_1	k_2						
0	0	29609	52.11	0	15534	52.23	2
-1	0	54961	50.04	0	29937	49.73	13
-1	1	82560	46.32	0	45085	45.87	15
-2	1	100886	45.11	0	58000	44.26	23
-2	2	121776	43.36	0	71564	42.31	30
-3	2	133306	42.66	0	82414	41.27	35
-3	3	147323	41.66	0	94248	40.05	44
-4	3	154240	41.21	0	103490	39.31	64
-4	4	162841	40.55	1	113536	38.45	73
-5	4	166979	40.23	1	121128	37.91	101
-5	5	172226	39.78	3	129372	37.28	114
-6	5	174931	39.53	3	135485	36.87	138
-6	6	178270	39.19	3	142254	36.38	157
-7	6	180074	38.99	3	147174	36.06	198
-7	7	182203	38.72	3	152623	35.68	215
-8	7	183437	38.57	3	156439	35.42	245
-8	8	184880	38.35	4	160888	35.11	270
-9	8	185874	38.21	4	163908	34.90	300
-9	9	187033	38.02	5	167408	34.66	335
-10	9	187795	37.91	5	169838	34.48	358
-10	10	188670	37.74	5	172576	34.28	400

Table V: Results for Airplane (512 x 512) and Tank (512 x 512) Grayscale Image

		Airplane (512×512)			**Tank** (512×512)		
Thresholds		**Max. Embedding Capacity (bits)**	**PSNR (dB)**	**Location Map Size (bits)**	**Max. Embedding Capacity (bits)**	**PSNR (dB)**	**Location Map Size (bits)**
k_1	k_2						
0	0	39557	52.00	0	16870	52.28	0
-1	0	63910	50.15	0	32050	49.75	0
-1	1	97798	46.55	0	47021	45.92	0
-2	1	111696	45.41	0	59928	44.31	0

-2	2	131125	43.79	0	74622	42.39	0
-3	2	139555	43.04	0	85874	41.34	0
-3	3	150605	42.11	0	98194	40.15	0
-4	3	155839	41.57	0	107301	39.41	0
-4	4	162115	40.95	0	118155	38.59	0
-5	4	165648	40.52	0	125795	38.05	0
-5	5	169532	40.07	0	134291	37.46	0
-6	5	172031	39.72	0	140244	37.04	0
-6	6	174534	39.37	0	147455	36.60	0
-7	6	176342	39.08	0	152306	36.27	5
-7	7	178168	38.78	0	157843	35.94	5
-8	7	179579	38.54	0	161591	35.67	8
-8	8	180941	38.29	0	165905	35.41	8
-9	8	182123	38.07	0	169013	35.19	14
-9	9	183196	37.86	0	172296	34.99	15
-10	9	184109	37.68	0	174736	34.82	16
-10	10	184937	37.49	0	177157	34.66	16

Table VI: Results for Baboon (512 x 512) and Peppers (512 x 512) Grayscale Image

		Baboon $(\mathbf{512 \times 512})$			**Peppers** $(\mathbf{512 \times 512})$		
Thresholds		**Max. Embedding Capacity (bits)**	**PSNR (dB)**	**Location Map Size (bits)**	**Max. Embedding Capacity (bits)**	**PSNR (dB)**	**Location Map Size (bits)**
k_1	k_2						
0	0	8053	52.29	0	20313	52.16	0
-1	0	15925	49.56	39	38912	49.84	133
-1	1	23791	45.62	40	58144	46.01	138
-2	1	31239	43.82	86	73661	44.55	552
-2	2	38846	41.78	86	89923	42.64	531
-3	2	45614	40.56	130	101948	41.74	1023
-3	3	52825	39.22	130	115654	40.56	1010
-4	3	58967	38.32	148	125096	39.96	1492
-4	4	65491	37.34	150	135913	39.15	1476
-5	4	70982	36.64	163	143160	38.73	1835
-5	5	76927	35.88	164	151513	38.16	1847
-6	5	81801	35.32	181	156587	37.85	2117
-6	6	87043	34.70	182	162905	37.43	2123
-7	6	91436	34.23	210	166707	37.21	2338
-7	7	96151	33.72	211	171302	36.89	2365
-8	7	100011	33.31	257	173988	36.72	2529
-8	8	104220	32.88	254	177299	36.47	2550
-9	8	107605	32.53	302	179222	36.33	2664
-9	9	111291	32.15	303	181634	36.14	2679
-10	9	114392	31.84	380	182996	36.03	2843
-10	10	117614	31.51	378	184771	35.87	2843

The performance of the proposed reversible scheme is also analysed for PSNR values with respect to different embedding capacity (in bits) are shown in Fig. 6.4 with maximum value of thresholds are taken as 10. The plots for sample images indicates EC (embedding capacity) of the scheme is very high with relatively low distortion.

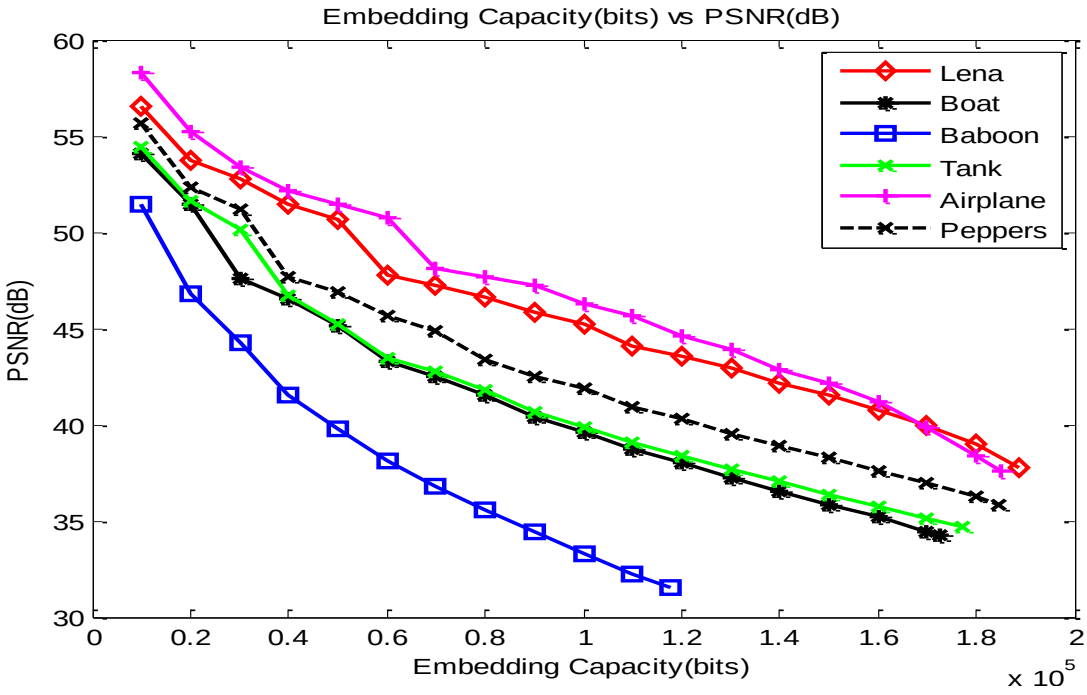


Figure 6.4: Embedding Capacity vs. PSNR

Fig. 6.5 shows the structural similarity (SSIM) index with respect to varying payload size. This figure shows that SSIM will remain above 92% for maximum embedding capacity, therefore it is very tough to distinguish between watermarked and host images visually by humans i.e., it is difficult that the existence of the watermark in the image is recognized by humans.

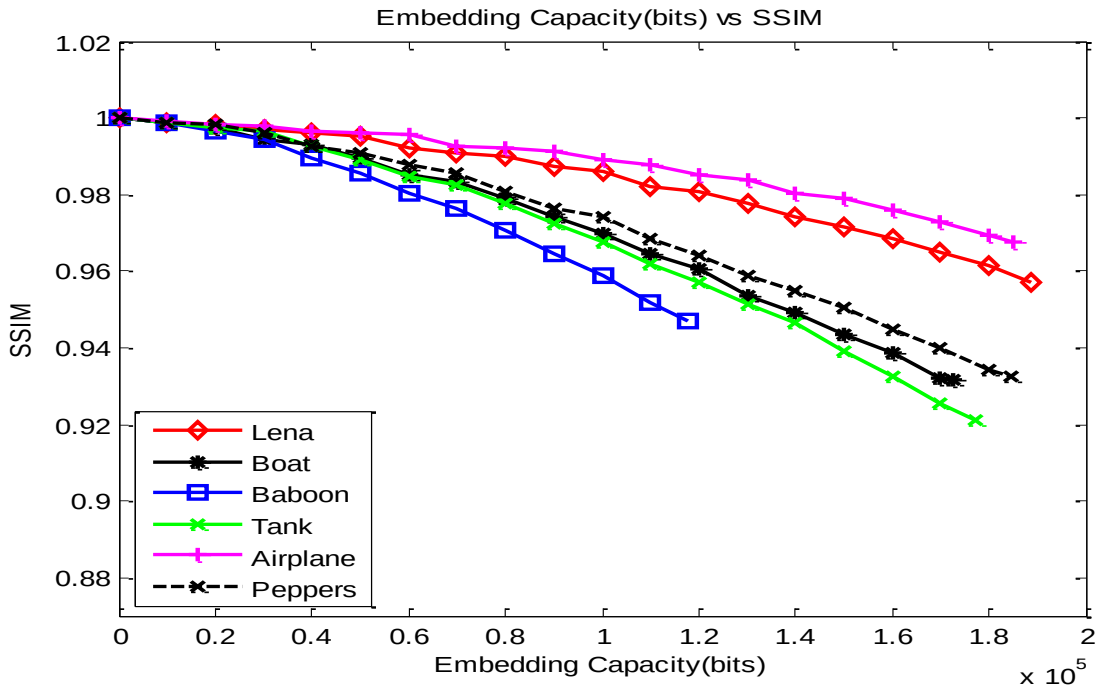
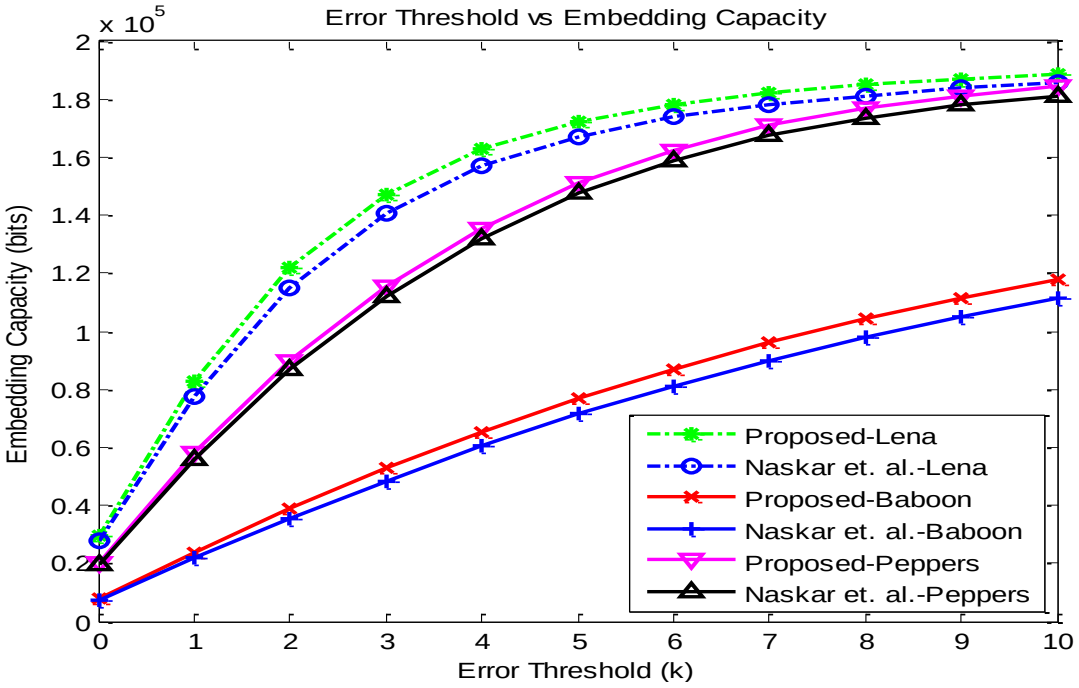


Figure 6.5: Embedding Capacity vs. SSIM

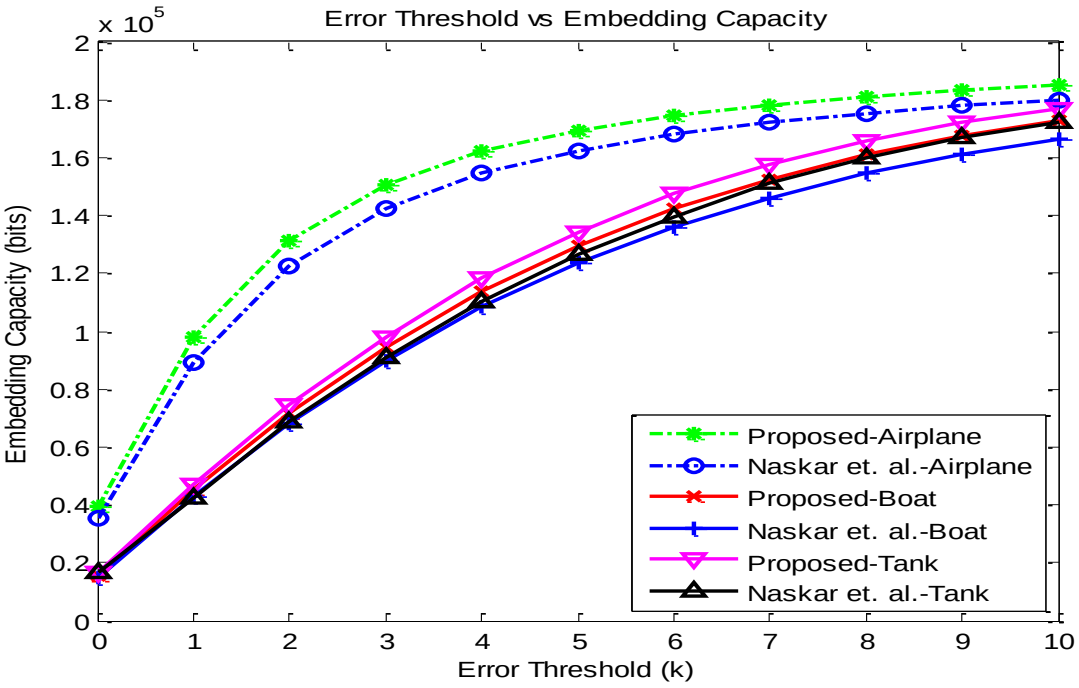
The proposed reversible watermarking algorithm described in chapter 4 and the scheme introduced in [20] were simulated in MATLAB 2013a for performance comparison. For comparing the proposed scheme with [20] in terms of maximum embedding capacity for different thresholds, error thresholds $k_1 = k_2$ is taken (as the author considered single threshold). Fig 6.6 shows comparison for maximum embedding capacity (in bits) with respect to fixed error thresholds, the plots for test images shows that the proposed reversible scheme has higher EC for different threshold values.

Further, comparison is made with some state-of-the-art reversible watermarking techniques for PSNR with varying payload size (in bits); results of comparison are indicated in Fig.6.7. Comparison results clearly demonstrate that the proposed scheme has enhanced performance.

The proposed reversible watermarking scheme also evaluated for some of the medical test images shown in Fig. 6.8 [39] of different sizes available in public domain. The results for medical test images for threshold values $k_1, k_2 \in [0,10]$ are given in Table below.

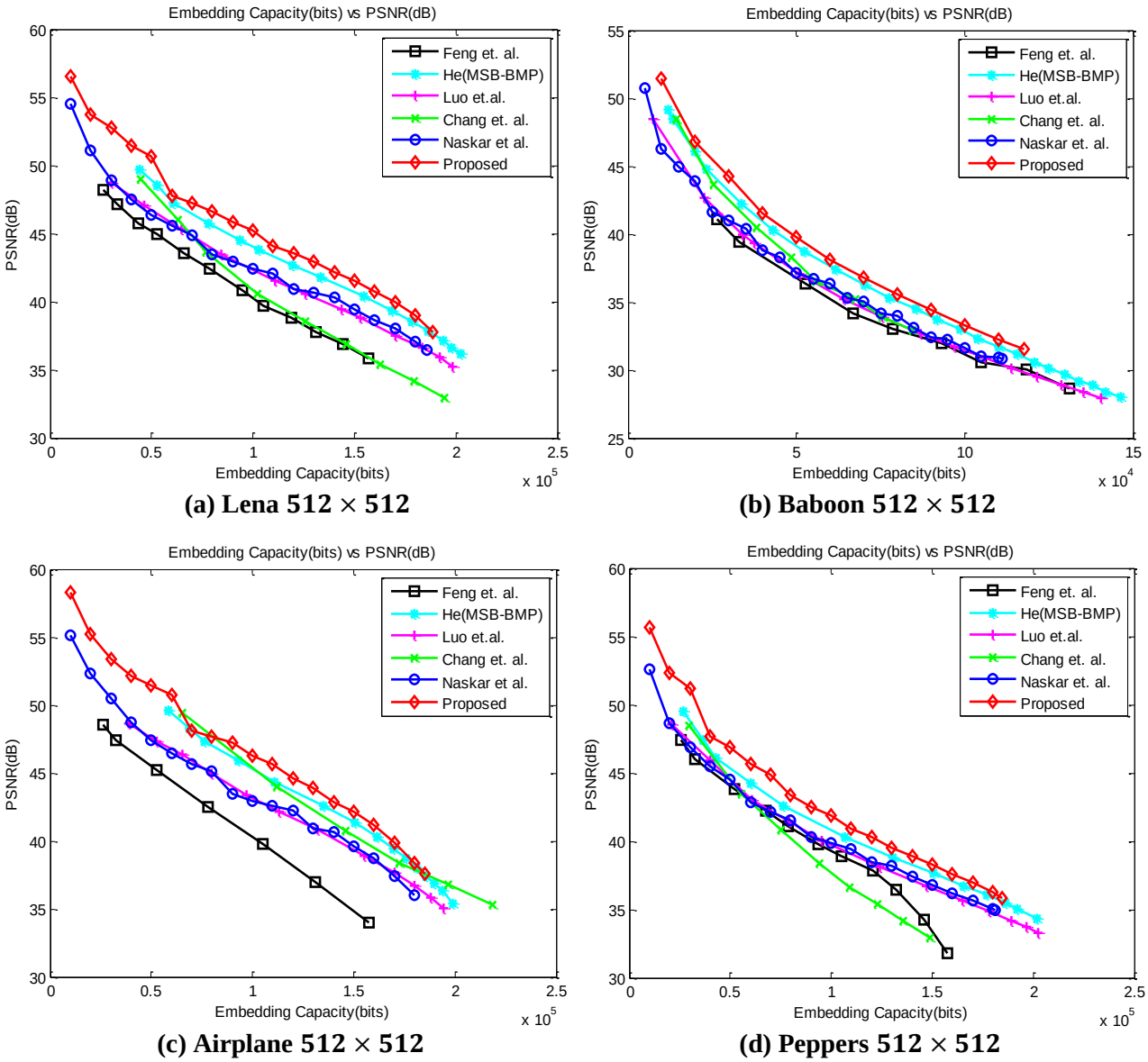


(a) Lena, Baboon and Peppers



(b) Airplane, Boat and Tank

Figure 6.6: Error Threshold vs. Maximum Embedding Capacity



(a) Lena 512 × 512 **(b) Baboon 512 × 512**

(c) Airplane 512 × 512 **(d) Peppers 512 × 512**

Figure 6.7: Performance Comparison with Other Schemes

(a) Ultrasound (597 × 895) **(b) Lungs (425 × 425)** **(c) Brain (911 × 699)** **(d) Fingerprint (512 × 512)** **(e) Heart (256 × 256)**

Figure 6.8: Medical Test Images

Table VII: **Results for Medical Test Images**

Test Image	Size	Maximum Embedding Capacity (bits)	PSNR (dB)	SSIM Index
Ultrasound	597×895	392537	40.7195	0.9727
Lungs	425×425	135120	45.0661	0.9854
Brain	911×699	459478	40.1135	0.9829
Fingerprint	512×512	157487	32.9725	0.9783
Heart	256×256	48991	42.8513	0.9781

6.2. Experimental Results and Analysis for Localized Tamper Detection

The localized tamper detection algorithm in digital images using the proposed reversible watermarking scheme described in chapter 5 was implemented in MATLAB for validation and testing. The effectiveness of the localized tamper detection algorithm was tested on sample images shown in Fig. 6.1. In experiments, block size taken as 32×32 and HMAC-SHA1 is used for generating the integrity check value for each block. The embedding results for different test images in terms of number of unit blocks, combined blocks, PSNR and SSIM index is shown in Table below. The results demonstrates that only few number of blocks are combined for HMAC-SHA1, thus produces small false rejection rate (FRR). Also for these images, the PSNR value greater than 35dB and the SSIM index is greater than 98% thus the structural information is preserved in watermarked image and the existence of watermark is not perceived visually.

Table VIII: **Embedding Results for Localized Tamper Detection**

Test Image	No. of 32×32 Unit Blocks	No. of Combined Blocks	PSNR (dB)	SSIM Index
Lena	256	0	48.154393	0.994429
Boat	256	0	43.972021	0.989381
Airplane	256	0	48.458800	0.995571
Tank	254	1	44.693777	0.989163
Baboon	238	9	35.859162	0.985730
Peppers	254	1	46.050849	0.990521

The performance of the localized tamper detection algorithm is also analysed for various types of attacks and corruption in marked image. The results shown in Fig. 6.9 demonstrates its effectiveness.

Figure 6.9: Performance Analysis of Localized Tamper Detection

(a) Crop Attack, (b) Copy Move attack, (c) Copy Paste Attack, (d) Random corruption

CHAPTER 7

CONCLUSION

In this project report, a three phase digital image reversible watermarking scheme using improved median based prediction is presented. The scheme utilizes the property of high relationship between nearby pixels in grayscale images and embed the watermark using prediction error expansion (PEE). Also, the scheme has high capacity with low distortion and size of side information is very small for most of the images. In addition, demonstrated that only single bit is sufficient for checking under/overflow that may occur when embedding is carried in watermarked pixel, thus construction of location map is more efficient than mentioned in [20]. The proposed scheme was simulated in MATLAB 2013a for analysis, validation and comparison. The simulation results for various standard and medical test images shows that the scheme has relatively high EC (embedding capacity) with relatively good PSNR and SSIM index value. An extensive experimental comparative study with some state-of-the-art schemes are also carried out which shows the advantages of the scheme.

Further, in this project I have also implemented localized tamper detection in digital image at receiver side as one of the application of the proposed reversible algorithm. Localization of tampered regions at the receiver side is achieved by dividing the image into small blocks and embedding the integrity value, which is computed by considering entire within block pixels along with block number. The scheme allows combining of current and next block dynamically, when the capacity of the block is not sufficient and an indicator bit is inserted in the image to facilitate the receiver. In addition, considering block number during integrity calculation provides protection against block copy and move type of attack. The experimental results shows that for test images the

number of blocks combined are relatively small with high PSNR and SSIM value of watermarked image. Also, results for different types of attacks on watermarked image demonstrates the effectiveness of the localized tamper detection scheme.

Future research would be directed towards increasing the embedding capacity with good PSNR & SSIM index value and reduction in size of location map. The embedding capacity can be increased by using better prediction methods or by using multiple layer embedding. Also, reversible watermarking for colour images using proposed scheme would be taken in future.

REFERENCES

[1] J. Saturwar and D.N.Chaudhari, "Digital Watermarking Scheme For Secret Images Using Visual Cryptography," *International Journal of Scientific & Engineering Research,* vol. 4, no. 6, June 2013.

[2] I. J. Cox, J. Kilian, F. T. Leighton and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions on Image Processing,* vol. 6, no. 12, December 1997.

[3] P. Singh and R. S. Chadha, "A Survey of Digital Watermarking Techniques, Applications and Attacks," *International Journal of Engineering and Innovative Technology (IJEIT),* vol. 2, no. 9, March 2013.

[4] A. Rashid, "Digital watermarking applications and techniques: a brief review," *International Journal of Computer Applications Technology and Research,* vol. 5, no. 3, pp. 147-150, 2016.

[5] S. Tyagi, H. V. Singh, R. Agarwal and S. K. Gangwar, "Digital watermarking techniques for security applications," in *International Conference on Emerging Trends in Electrical Electronics & Sustainable Energy Systems (ICETEESES), IEEE,* 2016.

[6] R. Naskar and R. S. Chakraborty, Reversible digital watermarking: Theory and Practices, vol. 5, Morgan & Claypool Publishers, pp. 1-130, 2014.

[7] N. R. Dasre, "On Watermarking Using Arnold and Wavelet Transform," *International Journal of Mathematics,* vol. 4, no. 5, 2016.

[8] R. Caldelli, F. Filippini and R. Becarelli, "Reversible Watermarking Techniques: An overview and a classification," *EURASIP Journal on Information Security,* vol. 2010, no. 1, p. 134546, 2010.

[9] W. He, Z. Cai and Y. Wang, "Flexible spatial location-based PVO predictor for high-fidelity reversible data hiding," *Information Sciences,* vol. 520, pp. 431-444, 2020.

[10] B. Shirisha and K. Prasad, "A Survey on Reversible Data Hiding Techniques," in *ICDSMLA 2019,* Springer, Singapore, 2020.

[11] Z. Ni, Y. Q. Shi, N. Ansari and W. Su, "Reversible data hiding," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 16, no. 3, pp. 354-362, 2006.

[12] C. De Vleeschouwer, J. E. Delaigle and B. Macq, "Circular interpretation of histogram for reversible watermarking," in *2001 IEEE Fourth Workshop on Multimedia Signal Processing (Cat. No. 01TH8564), IEEE,* Oct. 2001.

[13] M. U. Celik, G. Sharma, A. M. Tekalp and E. Saber, "Lossless generalized-LSB data embedding," *IEEE Transactions on Image Processing,* vol. 14, no. 2, pp. 253-266, 2005.

[14] G. Xuan, C. Yang, Y. Zhen, Y. Q. Shi and Z. Ni, "Reversible data hiding using integer wavelet transform and companding technique," in *International Workshop on Digital Watermarking*, Berlin, Heidelberg, Oct. 2004.

[15] N. A. Memon, A. Khan, S. A. M. Gilani and M. Ahmad, "Reversible watermarking method based on adaptive thresholding and companding technique.," *International Journal of Computer Mathematics,* vol. 88, no. 8, pp. 1573-1594, 2011.

[16] J. B. Feng, I. C. Lin, C. S. Tsai and Y. Chu, "Reversible watermarking: Current status and key issues," *IJ Network Security,* vol. 2, no. 3, pp. 161-170, 2006.

[17] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 13, no. 8, pp. 890-896, 2003.

[18] S. Weng, Y. Zhao, J. S. Pan and R. Ni, "A novel reversible watermarking based on an integer transform," in *IEEE International Conference on Image Processing*, Sep. 2007.

[19] H. J. Kim, V. Sachnev, Y. Q. Shi, J. Nam and H. G. Choo, "A novel difference expansion transform for reversible data embedding," *IEEE Transactions on Information Forensics and Security,* vol. 3, no. 3, pp. 456-465, 2008.

[20] R. Naskar and R. S. Chakraborty, "Reversible watermarking utilising weighted median-based prediction," *IET Image Processing,* vol. 6, no. 5, pp. 507-520, 2012.

[21] M. Ishtiaq, W. Ali, W. Shahzad, M. A. Jaffar and Y. Nam, "Hybrid predictor based four-phase adaptive reversible watermarking," *IEEE Access,* vol. 6, pp. 13213-13230, 2018.

[22] K. S. Kim, M. J. Lee, H. Y. Lee and H. K. Lee, "Reversible data hiding exploiting spatial correlation between sub-sampled images," *Pattern Recognition,* vol. 42, no. 11, pp. 3083-3096, 2009.

[23] L. Luo, Z. Chen, M. Chen, X. Zeng and Z. Xiong, "Reversible image watermarking using interpolation technique," *IEEE Transactions on Information Forensics and Security,* vol. 5, no. 1, pp. 187-193, 2009.

[24] M. A. M. Abadi, H. Danyali and M. S. Helfroush, "Reversible watermarking based on interpolation error histogram shifting," in *2010 5th International Symposium on Telecommunications, IEEE*, 2010.

[25] B. Feng, B. Yu, Y. Bei and X. Duan, "A reversible watermark with a new overflow solution," *IEEE Access,* vol. 7, pp. 28031-28043, 2018.

[26] F. Huang, X. Qu, H. J. Kim and J. Huang, "Reversible data hiding in JPEG images," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 26, no. 9, pp. 1610-1621, 2015.

[27] D. Hou, H. Wang, W. Zhang and N. Yu, "Reversible data hiding in JPEG image based on DCT frequency and block selection," *Signal Processing,* vol. 148, pp. 41-47, 2018.

[28] Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600-612, 2004.

[29] Z. Wang, E. P. Simoncelli and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, IEEE*, Nov. 2003.

[30] M. P. Turuk and A. P. Dhande, "A novel reversible multiple medical image watermarking for health information system," *Journal of Medical Systems*, vol. 40, no. 12, p. 269, 2016.

[31] X. Wu, J. Weng and W. Yan, "Adopting secret sharing for reversible data hiding in encrypted images," *Signal Processing*, vol. 143, pp. 269-281, 2018.

[32] X. Zhang, Z. Qian, Y. Ren and G. Feng, "Watermarking with flexible self-recovery quality based on compressive sensing and compositive reconstruction," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 4, pp. 1223-1232, 2011.

[33] J. Dittmann, A. Steinmetz and R. Steinmetz, "Content-based digital signature for motion pictures authentication and content-fragile watermarking," in *Proceedings IEEE International Conference on Multimedia Computing and Systems, IEEE*, 1999.

[34] X. Qi and X. Xin, "A singular-value-based semi-fragile watermarking scheme for image content authentication with tamper localization," *Journal of Visual Communication and Image Representation*, vol. 30, pp. 312-327, 2015.

[35] R. Naskar and R. S. Chakraborty, "A generalized tamper localization approach for reversible watermarking algorithms," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 9, no. 3, pp. 1-22, 2013.

[36] J. M. Turner, "The keyed-hash message authentication code (hmac)," Federal Information Processing Standards Publication, 198-1, 2008.

[37] "CVG–UGR Image Database Computer Vision Group, University of Granada," [Online]. Available: http://decsai.ugr.es/cvg/dbimagenes and http://decsai.ugr.es/cvg/CG/base.htm.

[38] "Sample Images," [Online]. Available: http://eeweb.poly.edu/~yao/EL5123/ SampleData.html.

[39] "Test Images Collections," [Online]. Available: https://www.hlevkin.com/06 testimages.htm.