An adaptive quad tree based transform domain steganography for textual data

Dissertation submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Information Security

Submitted by

Jobanjeet Kaur
(801533010)

Under the supervision of

Dr. Shreelekha Pandey
Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA - 147004

July 2017

Certificate

I hereby certify that the work, which is being presented in the thesis, entitled *An adaptive quad tree based transform domain steganography for textual data*, in partial fulfillment of the requirements for the award of the degree of *Masters of Engineering* in *Information Security* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Shreelekha Pandey* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Jobanjeet Kaur
(801533010)

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

14/7/14

Dr. Shreelekha Pandey
Assistant Professor
Computer Science and Engineering Department
Thapar University

Acknowledgements

No volume of words is enough to express my gratitude towards my supervisor, **Dr. Shreelekha Pandey**, Assistant Professor, Computer Science and Engineering Department, Thapar University, Patiala, who has been very concerned and has overseen the work presented in this thesis report. She has helped me to explore this vast field in an organised manner and provided me with all the ideas on how to work and systematically proceed towards a research oriented venture.

I am also thankful to **Dr. Maninder Singh** (Head, Computer Science and Engineering Department, Thapar University, Patiala) and **Dr. Shreelekha Pandey** (PG Coordinator), for the motivation and inspiration that triggered me for the thesis work.

I would also like to thank the staff members and my colleagues who were always there in the need of the hour and provided with all the help and facilities, which I required, for the completion of my thesis.

Most importantly, I would like to thank my parents, friends and the almighty for showing me the direction out of the blue, to help me to stay calm in the oddest of the times and keep moving even at times when there was no hope.

Jobanjeet Kaur
(801533010)

Abstract

Internet is most popularly used for communication. However, most of these communications are confidential and must not be detected by intruders. Steganography, a well-known traditional approach, provides an efficient way to obscure the presence of such a communication by hiding data in media like images, audio, video, etc. Also, today's world communicates mostly via images which are known to have very high level of redundancy. As a result, image steganography seems to be the best suitable choice among all the available variants. This manuscript thus presents a block based transform domain image steganography approach to embed a secret text message within a cover image.

The approach is designed by combining the concept of quad tree blocks with discrete cosine transform (DCT) which is a transform domain steganography technique. A quad tree decomposition of cover image results in variable sized square blocks. The presented approach adaptively determines locations within variable sized quad tree blocks in a cover image for embedding the secret text. The number of blocks obtained for a cover image is controlled using a threshold and a minimum block size. Results are generated for various combinations of these two values and are analyzed on the basis of capacity, peak signal to noise ratio (PSNR), and structural similarity (SSIM) index. The obtained results prove the effectiveness of the presented scheme over the existing spatial and transform domain techniques (without block and fixed block based) in terms of all the considered parameters. In addition, a visual examination further proves that the presented approach provides good imperceptibility of the secret message, thus hiding the presence of any secret communication.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

bpp	bits per pixel
AMBTC	Absolute Moment Block Truncation Coding
AR-DCT	Adaptive Region Discrete Cosine Transform
AVI	Audio Video Interleaved
CISSKA	Color Image Steganography using Stego Key directed Adaptive LSB
CLSB	Classic Least Significant Bit method
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DWT	Discrete Wavelet Transform
EBE	Edges Based data Embedding
EMD	Exploit Modification Direction
FPDPSK	Fold Phase distribution Differential Phase-Shift Keying
GAR	Global Adaptive region
HVS	Human Visual System
ICMP	Internet Control Message Protocol
IP	Internet Protocol
JPEG	Joint Photographic Experts Group
LSB	Least Significant Bit
LWT	Lifting Wavelet Transform
MBDH	Multi-plane Block Data Hiding
MLEA	Multi-Level Encryption Algorithm
MPEG	Moving Picture Experts Group
MSE	Mean Squared Error
NBSPC	Neighbour Block Signal Phase Comparison
OSI	Open System Interconnection
PIT	Pixel Indicator Technique
PSNR	Peak Signal to Noise Ratio
PVD	Pixel Value Differencing
PXORSS	Probabilistic XOR Secret Sharing
SCC	Stego Color Cycle
SKA-LSB	Stego Key directed Adaptive Least Significant Bit
SPIHT	Set Partitioning in Hierarchical Trees
SSIM	Structural SIMilarity
ST-FMM	Steganography Five Modulus Method

TCP	Transmission Control Protocol
TLEA	Two Level Encryption Algorithm
UDP	User Datagram Protocol
VOIP	Voice over IP

Chapter 1

Introduction

Steganography is a practice of concealed writing in which secret data is hidden in a cover media so as to establish a secret communication. Any medium used for information sharing, like images, text, audio and video files, can successfully be used as a cover media. These information sharing mediums are known to possess high degree of redundancy and the redundant areas can simply be used for hiding information [4]. These days Internet is popularly being used for sharing several types of information, ranging from a trivial file to an important document, particularly in the form of an image. Also considering human visual perception, it's convenient to hide data in redundant image areas as compared to other available carriers. An image used as a cover media in a steganographic system is popularly called a cover image. The secret message is embedded in the cover image to obtain a stego image and an associated stego key. Stego key is needed to retrieve the hidden message.

The chapter presents the motivation and objectives of this manuscript in addition to the basics of steganography. Starting with the motivation in Section 1.1, the objectives focused in this work are listed in Section 1.2. The history of steganography, its mechanism and types of steganography techniques are subsequently discussed in Section 1.3, Section 1.4, and Section 1.5, respectively. The steganography taxonomy based on the cover medium utilized is given in Section 1.6 and other techniques closely related to steganography are briefed in Section 1.7. Lastly, the organization of the thesis is specified in Section 1.8.

1.1 Motivation

Important factors that govern data hiding in images are security, image quality, and capacity. Security signifies inability of attackers to discover the secret message, capacity implies the number of bits that can be embedded, and image quality stands for the amount of distortion in the stego image with respect to the original

image. Need for higher capacity leads to more distortion but higher imperceptibility forces lower capacity.

Balance among security, image quality, and capacity is a critical issue which is to be dealt properly. Several researches tried to equalize these factors by dividing an image into fixed blocks before embedding the data [5, 6, 7, 8]. However, natural images are not statistically stationary over the fixed block region and a quad tree adaptive region (QTAR) embedding scheme is found to work better in such cases [9]. Using this property, secret images are embedded within a cover image which is first divided into variable blocks. An increased embedding capacity along with imperceptibility of stego images is observed by means of this approach.

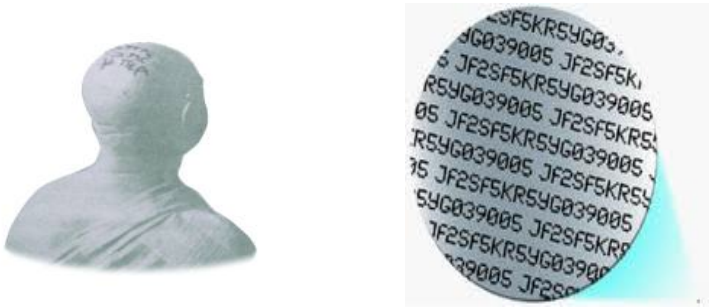
1.2 Objectives

The approach presented in this work attempts to explore quad tree based method to embed text in cover images utilizing the fact that high frequency DCT coefficients do not contribute in representing image correlation and thus can easily be used for embedding. In view of this, following are the objectives of this dissertation:

- To study transform domain steganography using discrete cosine transform (DCT).
- To study and implement quad tree decomposition on color images.
- To implement a steganographic system that embeds a secret text message in high frequency DCT regions of quad tree blocks within a color image.
- To analyze and compare the obtained results with other spatial and transform domain techniques (with or without block based).

1.3 A brief history

Steganography has been used since historical times by military and political leaders. One of the story narrated by Herodotus describes a technique followed by a nobleman of Medea, who hid message in the belly of an unskinned hare and a messenger disguised as a hunter delivered it [10]. Another story is about a Persian nobleman Histiaeus who shaved a slave's head and tattooed the message on his scalp, an example is shown in Fig. 1.1a. The slave then grew his hair and was dispatched to his destination, where his head is re-shaved to reveal the message.



(a) (b)

Figure 1.1: Historic steganography habits to hide secret message [1]. (a) Message on a shaved head. (b) Enlarged view of a microdot.

In one more incidence, information about the attack on Greece was hidden on a wax tablet. First wax melted from the tablet, and then message was inscribed on the underlying wood. Finally, wax was reapplied on the wood. After that tablets were transported without anyone knowing about the existence of such a message within the wax tablet.

In World War II, spy used to write messages on a paper using invisible inks prepared from fruit juices and milk. The recipient had to expose the paper toward warmth and revealed the written message. A Nazi spy, George Dash used a special solution of copper sulfate to write messages on his handkerchief. These types of invisible inks were more sophisticated and required the application of chemical substances, like ammonia fumes to reveal the message.

A "microdot" technique, known as "the enemy's masterpiece of espionage" was also developed by Nazis [11]. An example is shown in Fig. 1.1b. Microdot was a text or photograph whose size was same as that of a typed period, which made their presence unnoticeable by an inspector. When developed, microdots were reproduced to typewritten images of standard sizes with good clarity. The Germans extensively used these microdot for secret transmission of technical drawings and other printed data.

During same time, the US marines encrypted their radio messages with the help of Navajo Indians also known as "code talkers". Navajo Indians used their native language for encryption and decryption. At that time only 28 non-Navajos could understand and speak the language, none of whom were Germans or Japanese. Later, a slang version of the language was utilized that was unidentified even by the Navajo speakers. Similarly in 1968, members of USS Pueblo (AGER-2) intelligence ship, which was held captive by North Korea during the cold war, used sign language to inform the United States about their confinement [1]. They did so during the staged photo opportunities thus making the North Koreans

imperceptible to the message.

The techniques and incidents discussed in this section are just a few historical examples in the domain of steganography, but had played significant role in setting the benchmark of stagenography being used today.

1.4 Steganography mechanism

Fig. 1.2 demonstrates a block diagram of a general mechanism for steganography. At the sender end, a cover medium is embedded with the secret message using a key or a password, known as a stego key. Depending upon the type of steganography technique employed, the stego key may or may not be shared with the receiver. The cover medium embedded with the secret message is termed as a stego medium which will be transferred over the communication channel. Lastly at the receiver end, the embedded secret message is extracted from the stego medium using a stego key.

It is really important that the stego medium should appear identical to the cover medium, i.e. they should be indistinguishable. Several measures are proposed in literature to compute the degree of similarity between the cover and stego mediums [12]. In this work the popular Peak signal-to-noise ratio (PSNR) and Structural SIMilarity (SSIM) index are used as measures of similarity [13].

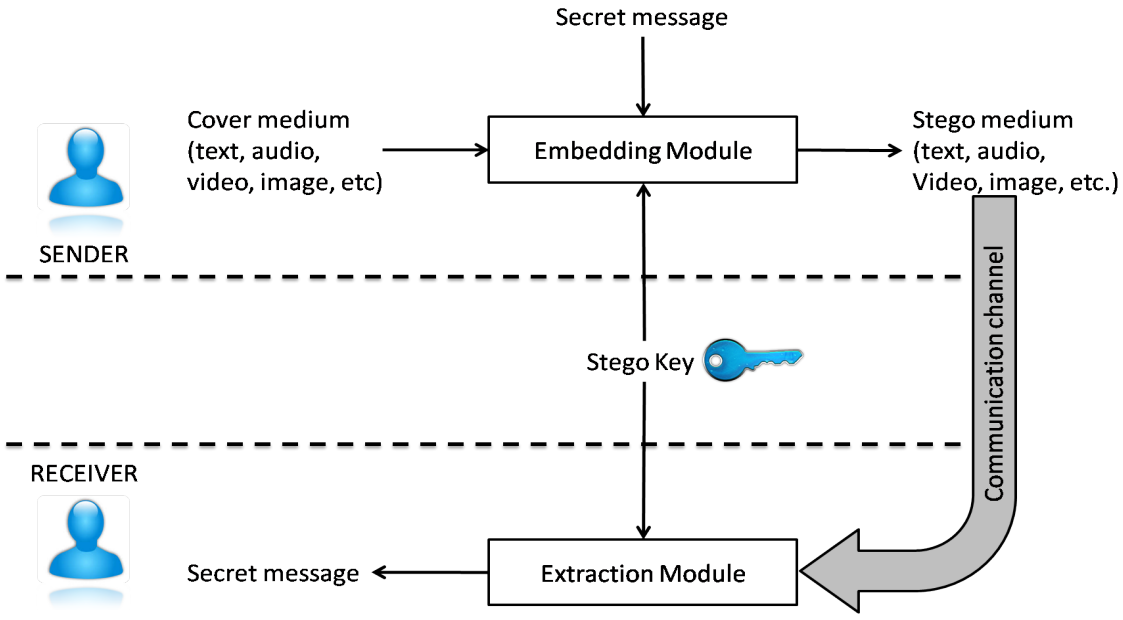


Figure 1.2: A block diagram showing steganography mechanism.

1.5 Types of steganography techniques

The three basic types of steganography techniques are as follows [12].

1. Pure steganography,
2. Public key steganography, and
3. Secret key steganography

1.5.1 Pure steganography

Pure steganography does not require a prior exchange of any information (like a stego key) before sending the actual message. The block diagram shown in Fig. 1.3 depicts pure steganography. The security of such systems depends entirely on their secrecy. Pure steganographic system can be defined by the quadruple $S = (C, M, E, D)$, where

C is the set of all possible covers,

M is the set of all possible secret messages being $\mid C \mid \geq \mid M \mid$,

$E : C \times M \longrightarrow C$ is the embedding function, and

$D : C \longrightarrow M$ is the extraction function satisfying the property that $D(E(c, m)) = m$, for all $m \in M$ and $c \in C$.

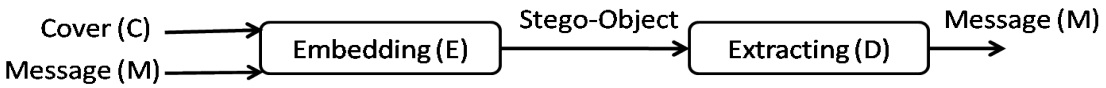


Figure 1.3: Pure steganographic system [2].

1.5.2 Secret key steganography

Secret key steganography is comparable to a symmetric cipher system. This type compromises with the objective of performing secret communication as it requires prior exchange of keys. Usually some characteristics of the cover medium are hashed so as to simplify the exchange. But if the steganographic system alters cover medium characteristics, then the stego key cannot be regenerated at the reception end. The block diagram for this steganography type is shown in Fig. 1.4. Formula definition of secret key steganographic system is given by quintuple $S = (C, M, K, E, D)$, where

C is the set of possible covers,

M is the set of possible secret messages being $\mid C \mid \geq \mid M \mid$,

K is the set of possible keys,

$E_K : C \times M \times K \longrightarrow C$ is the embedding function, and

$D_K : C \times K \longrightarrow M$ is the extraction function satisfying the property that $D_K(E_K(c, m, k), k) = m$, for all $m \in M$, for all $c \in C$, and for all $k \in K$.

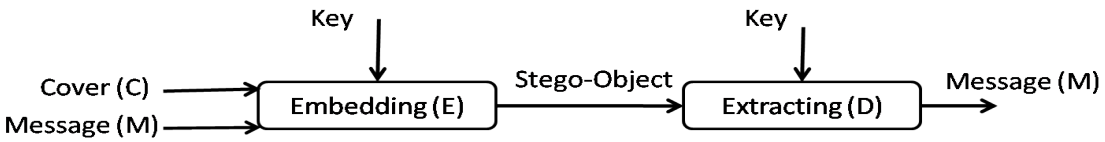


Figure 1.4: Secret key steganographic system [2].

1.5.3 Public key steganography

Public key steganography does not require a prior exchange of keys instead it works with two keys which are used in a manner similar to public key cryptography. One key is public and the other is private. The public key is stored in a public database and is used during embedding. The private key, also known as secret key, is known only to the receiver, which is used for extracting the message from the stego medium. The block diagram of public key steganographic system is shown in Fig. 1.5.

Public key steganographic systems can be constructed using an equivalent cryptographic system. The sender and the receiver first exchange public keys before participating in any communication. The sender encrypts the message with the receiver's public key and then embeds the encrypted message in a cover medium. This system believes that the encrypted message is random enough and can effortlessly be hidden in plain sight. Sometimes the cover medium does not contain the secret message, but still the extraction function gets executed without any fault. In case the message is present then extracted text is the required secret message.

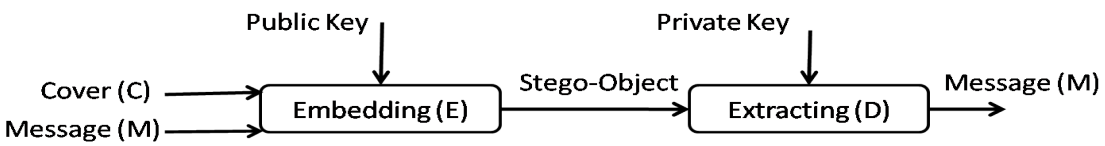


Figure 1.5: Public key steganographic system [2].

1.6 Classification of steganography based on cover medium

Any means of information sharing can be used as a cover media and depending upon the one utilized, steganography can be categorized as follows:

1. **Image steganography**. It uses an image as a cover medium and pixel intensities for storing secret information. Human visual system cannot detect any changes in luminance of color vectors; image steganography exploits this weakness during embedding.

2. **Network steganography**. It uses network protocol like transmission control protocol (TCP), internet control message protocol (ICMP), user datagram protocol (UDP), internet protocol (IP), etc. as a cover medium [14]. It utilizes unused header bits in open system interconnection (OSI) network layer model for sending secret information.

3. **Video steganography**. It uses video which is actually a combination of images, as a cover medium. Similar to images, any change in the frames of video is usually imperceptible to human visual system. A few commonly used video formats are moving picture experts group (MPEG), audio video interleaved (AVI), MPEG-4 (Mp4), etc.

4. **Audio steganography**. It uses an audio file as a cover medium. A digitized secret message is embedded in an audio signal by altering its binary sequence. Prevalence of voice over IP (VOIP) make audio files a significant medium these days. It uses digital audio formats like AVI, MPEG, etc.

5. **Text steganography**. It uses text as a cover medium. Certain characteristics of a text or formatting options like number of tabs, capital letters, white spaces, etc. are used to hide information. Text steganography hides the message such that its presence is undetectable and its extraction is possible even in the presence of noise.

1.7 Other related techniques

Watermarking deals with protecting intellectual property. Generally in watermarking technique, a signature is hidden that signifies the ownership of an object for the sake of copyright protection. On the other hand, **fingerprinting** supplies different copies of objects embedded with unique marks to the customers. This

helps in identifying customers who break their licensing agreement. Watermarking and fingerprinting hide data and this fact is usually known to public. Sometimes they are even visible. But in steganography, imperceptibility of hidden data is very important. In fingerprinting or watermarking, a successful attack is identified as a removal of the secret mark while in steganography, the attack is successful if adversary detects the hidden data.

Another technique that protects data from illicit access of unwanted entities is **cryptography**. It encrypts data with the help of a key and transforms data into another form thus ensures protection. The key is either a secret between the intended parties or is conveyed to the receiver as it is needed to decrypt the encrypted data. However, the encrypted data is plainly visible and attracts attention towards it. Thus, some countries have prohibited or limited cryptography's usage as they would not be able to gain intelligence with the encrypted data. Steganography differs from cryptography significantly. Cryptography hides only the secret message, but steganography hides the existence of any such message. This way, steganography does not attract attention towards the message as an object of interest. However, researchers do combine cryptography with steganography, i.e. first encrypt the secret data and then hide it in a cover media [15]. This combination provides an added security by allowing communication between intended parties only and no other party even knows the existence of such a communication.

1.8 Organization of the thesis

The rest of the thesis is organized as follows:

Chapter 2 presents a literature review of spatial and transform domain image steganography techniques.

Chapter 3 introduces the mathematical requisites and discusses the steganographic framework presented in this manuscript. It details both the phases, embedding and extraction, and then illustrative examples are shown.

Chapter 4 describes the parameters utilized to analyze the performance of the presented block based transform domain steganographic techniques. Comparisons with several spatial and transform domain techniques may or may not utilizing the concepts of blocks are also presented.

Chapter 5 concludes the work presented in this dissertation along with the summary and a quick look at the future of this work.

Chapter 2

Literature Review

This work uses images as a cover medium, thus a literature related to the sphere of image steganography is explored first for proper understanding. Image steganography approaches are broadly classified under the heads spatial domain and transform domain. Both the domains are well explored in literature and the relevant works in each of the two domains are included in this chapter. Based on the discussion a summary is also presented.

The contents in this chapter are organized as follows. Section 2.1 discusses various spatial domain image steganography approaches and Section 2.2 contains the literature under the category of transform domain image steganography approaches. Section 2.3 presents a summary and a few observations made during the study.

2.1 Spatial domain approaches

Spatial domain steganography approaches deal with direct manipulation of pixels. A few well-known spatial domain approaches are least significant bit (LSB) [16, 17, 18], pixel value differencing (PVD) [19, 20], edges based data embedding (EBE) [21, 22], and pixel pair matching [23]. LSB substitution is the most basic method in this category. Being a straightforward approach, LSB is vulnerable to simple attacks. This weakness is overcome by an enhanced version of LSB replacement known as stego color cycle (SCC) [24]. It embeds message in three color channels of the chosen cover image following a predetermined cyclic order. At a time single color channel is used for embedding till the message finishes or all image pixels are used. The order followed is R, G, and B, i.e., Red channel of the 1^{st} pixel contains 1^{st} message bit, Green channel of 2^{nd} pixel has 2^{nd} message bit, Blue channel of 3^{rd} pixel contains 3^{rd} message bit, and so on. In this way, message gets disbursed in three color channels, making it better than simple LSB. However, SCC too is prone to attacks as it follows a fixed cyclic order. Also the payload size, i.e.

Table 2.1: Embedding policy used in pixel indicator technique (PIT) [3].

Indicator channel (Intermediate LSB, LSB)	Data channels	
	Channel 1	Channel 2
00	No embedding	No embedding
01	No embedding	2 bits embedded in 1-LSB and 2-LSB
10	2 bits embedded in 1-LSB and 2-LSB	No embedding
11	2 bits embedded in 1-LSB and 2-LSB	2 bits embedded in 1-LSB and 2-LSB

capacity of SCC is small.

A pixel indicator technique (PIT) is thus developed to enhance the payload size [3]. The two LSBs of any single color channel (R, G, or B) are used as an indicator of data stored in the two LSBs of the remaining two channels. The possible values of an indicator channel used to decide the embedding policy are shown in Table 2.1. The approach uses R, G, and B color channels as indicators, respectively for first, second, and third pixel of an image. This order is then repeated for rest of the pixels. PIT results in high embedding capacity, but the improvement is negligible if large number of indicator channels has 00 LSB value which signifies no embedding. Another approach tries to improve the security of a simple LSB method by adding the requirement of a secret key to extract the embedded data [25]. Although it provides the same embedding capacity as that of LSB i.e. one bit per pixel (bpp). Working on the same lines, a steganography five modulus method (ST-FMM) divides the cover image into square blocks and disperses the message among them [26]. This distribution boosts the difficulty of extraction process thus improving security. But success of ST-FMM is limited because of an existing dependency between the capacity and size of the square block. Overall ST-FMM improves security but at a cost of reduced capacity.

A semi-reversible approach combining the concepts of interpolation and k-LSB substitution is proposed to hide an image or a video [18]. Before actual embedding the cover image is scaled up using interpolation and later it is scaled down. The scaling not only increases capacity, but also guarantees to maintain a high visual quality of stego image. The experimental results report 37.54 dB and 43.94 dB PSNR for $k = 3$ and $k = 2$, respectively. A reversible approach based on exploiting modification direction (EMD) using two images is proposed [27]. Firstly two visually similar images are generated from a chosen cover image and data is

embedded in the identified pixel pairs. During data embedding, gray pixel values of first image are modified by up to one level using traditional EMD method. Afterwards second image pixels are modified adaptively with reference to the first image. The results show a satisfactory quality of stego image and high secret data hiding capacity.

A unique smooth or complex block steganography method based on absolute moment block truncation coding (AMBTC) is devised [6]. It is characterized by minimum distortion, low computational complexity, and high payload capacity. Secret data is embedded in blocks which are identified using a predefined threshold. For smooth blocks, bit planes are used for embedding followed by two levels of quantization that minimizes the distortion. In case of complex blocks, the order of two quantization levels is exchanged by toggling bit plane to hide secret bits. Existence of adjustable threshold makes this scheme flexible and ensures its relevancy in varied applications.

A magic LSB substitution method (M-LSB-SM) combining a multi-level encryption (MLE) with spatial domain steganography is developed for RGB images [28]. The cover image is first converted into hue saturation intensity (HSI) space, then based on the I-plane four equal sized sub-images are formed which are rotated at different angles depending on the secret key. The data is also divided in four parts, encrypted using MLE, and then each part is embedded individually into four rotated sub-images using LSB substitution. The results show enhanced visual quality, good imperceptibility and multiple levels of security. Another secure steganographic framework relies on encryption so as to make extraction difficult for adversaries [29]. It is based on stego key directed adaptive LSB substitution (SKA-LSB). Stego key and secret data both are encrypted using a two-level encryption algorithm (TLEA) and MLE, respectively. Then depending on R channel, secret key, secret data, and MLE, the LSB is determined adaptively to embed data in the cover image. This framework proves to be very effective as it is able to balance the security and image quality pretty well, along with less computational complexity and reasonable payload. A data hiding method utilizing a truth table of exclusive-OR (XOR) operator in deoxyribonucleic acid (DNA) is also developed for color images [30]. The DNA-XOR truth table is applied to evaluate input values and then select those that lead to the highest PSNR for the embedding process. It also encrypts data using the cover image as a key. The data is first divided into three secret shares which are embedded individually in R, G, and B channels of the cover image. The success of this method is confirmed by the results obtained during comparisons.

2.2 Transform domain approaches

Transform domain steganography approaches use domain coefficients to embed data. Popular transform domain approaches include discrete fourier transform (DFT) [31], discrete cosine transform (DCT) [32, 33], and discrete wavelet transform (DWT) [34, 35]. An approach to hide a secret image within a cover image is developed using a concept from the domain of digital communication, mainly differential phase-shift keying (DPSK) [36]. Secret image is first compressed using set partitioning in hierarchical trees (SPIHT) method. SPIHT not only reduces the number of bits but also helps in high quality reconstruction. The location of embedded data is found using neighbor block signal phase comparison (NB-SPC). Lastly, image quality is improved using a fold phase distribution DPSK (FPDPSK). This approach provides better quality as well as noise margin as compared to DPSK on same test conditions.

One work focuses on increasing the embedding capacity but with minimum changes to cover image which is in JPEG format [37]. It devised a DCT-M3 algorithm that uses a modulus 3 difference of two DCT coefficients to embed two compressed secret message bits. In this way, the embedding process makes minimal changes to the cover image. The approach not only improves the capacity by approximately 16.7% but also ensures protection against blind steganalysis approaches. The knowledge of DC coefficients is also explored to hide secret bits in 1-LSB and 2-LSB sequentially [8]. The approach utilizes either low or middle frequency. However, higher embedding capacity, less visual distortion and hence better PSNR as well as security is observed with middle frequency. Thus, a tool to hide confidential information of nuclear reactors is proposed based on this methodology using middle frequency.

A novel scheme explores a combination of chaotic map with DCT as well [32]. First DCT is applied over the cover image followed by zigzag scanning of AC coefficients starting from the least significant to the most significant one. Chaotic function is used during scanning to get embedding positions and a threshold on SSIM is employed to determine the maximum allowed capacity. This makes the approach flexible and helps in the generation of an imperceptible stego image. Another work discusses a fast, robust, efficient, and flexible chaotic approach based on 3-dimensional chaotic cat map with lifted DWT [34]. Irregular cat map outputs are used for embedding secret message. For improved robustness, Sweldens' lifting scheme is applied as it ensures integer transforms. The empirical results prove acceptability of the approach in terms of security and imperceptibility. A new algorithm providing high visual quality and embedding capacity is developed in

lifting wavelet transform (LWT) domain [38]. It uses low-high, high-low, high-high pass filter bands as cover images. Firstly, secret data is divided into three shares using probabilistic XOR secret sharing (PXORSS) scheme, the pieces with least distorted probabilities are then embedded in high-pass filter bands as secret share of the cover image.

A range of works have proposed the embedding of secret data in noisy image areas. This idea helps in reduced distortion, but lowers the embedding capacity too as some of the areas remains unused. A work tries to handle this issue by means of an adaptive multiple bit-planes image steganography using block data hiding (MPBDH) [39]. This approach identifies complex regions in the cover image by utilizing multi-bit planes along with the computation of adaptive complexity threshold. The obtained results show an increase in embedding capacity as well as security over the earlier approaches based on block and pixel complexity. Another work explores the relationship between an image quality and data hiding capacity limits by proposing an adaptive-region DCT (AR-DCT) data hiding scheme [7]. The results show improvements in embedding capacity and maintaining the image quality. The approach remarkably achieves a capacity of 20 bpp which is higher than any other existing approaches.

Among transform domain approaches, DCT is more preferable. A group of researchers explored it to promote optimal embedding capacity while enhancing the image quality and imperceptibility for color images [5, 9]. One of the works employs the idea of global adaptive region (GAR) embedding which identifies a region in each fixed sized DCT block. The region size is adaptive as it is depends on the amount of correlation in the pixel values within a block. Another work divides the cover image using a quad tree adaptive region (QTAR) scheme. It utilizes statistical characteristics of an image to decide the segments according to correlation of pixels. GAR enhances the capacity and imperceptibility as compared to other spatial and DCT based steganographic approaches. But the one using QTAR supersedes the fixed block based adaptive region scheme and thus results in the best performance.

2.3 Summary

Spatial domain approaches directly modify the pixel values of a cover image while transform domain approaches use transform coefficients to hide a secret message. Ability to use less significant areas for hiding makes transform domain approaches more preferable for robust communication [40]. Another advantage associated

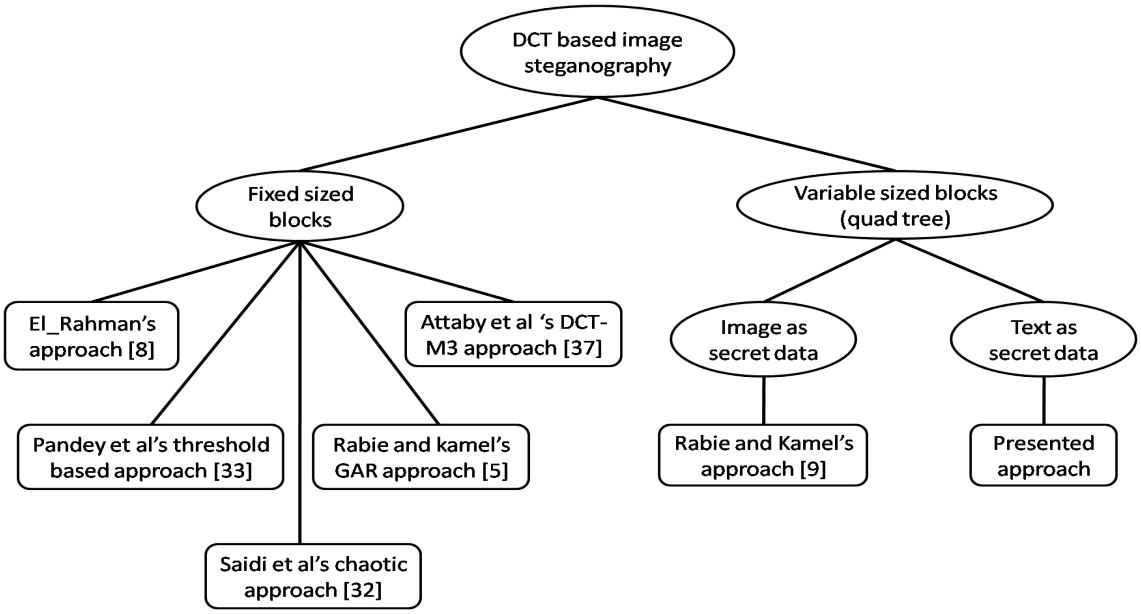


Figure 2.1: Categorization of research areas in DCT based image steganography.

with transform domain approaches is image format independence, this further eases data embedding in any format of choice.

Researchers have tried to maintain a proper balance among factors, like security, image quality, and capacity, by dividing an image into fixed or variable blocks. It is observed that the concept of blocks is more commonly used in transform domain steganography approaches, particularly DCT. Fig. 2.1 shows the categorization of research works in DCT domain. Clearly, most of the methods focus on fixed blocks. To the best of our knowledge only one work has utilized quad trees to get variable blocks on a cover image for embedding the secret image.

Observing all the facts, this work attempts to explore the applicability block based transform domain steganography approaches in embedding textual data, particularly a QTAR based variable sized block approach.

Chapter 3

Methodology

The steganography approach explored in this work lies in the category of block based transform domain image steganography. For proper understanding of the presented methodology, knowledge of discrete cosine transform (DCT) and quad trees concepts are necessary. Thus the objectives focused in this chapter include the study of DCT and quad tree decomposition. After this design of the steganographic system is followed using examples.

The contents in this chapter are organized as follows. The required concepts of DCT and quad tree are included in Section 3.1. Section 3.2 discusses the general framework of the presented steganographic system. Workings of embedding and extraction processes are detailed in Section 3.3 and Section 3.4, respectively. An illustrative example for images with sizes 256×256 and 512×512 is discussed in Section 3.5. Lastly, Section 3.6 presents the chapter summary.

3.1 Mathematical background

The concepts of quad trees and DCT are extensively explored with respect to images in several studies as they play a significant role in image compression. In addition, quad trees are found suitable for segmentation and smoothing operations. This section gives a brief introduction of both these notions highlighting their applicability in the presented steganography approach.

3.1.1 Quad tree decomposition

Quad tree is a tree-based data structure in which each non-leaf node has four children. Region quad tree, a commonly used variant of quad trees, partitions a two dimensional space into equal sized quadrants, then into sub-quadrants, and so on. In region quad trees, each leaf node contains data corresponding to a sub-region.

The level of refinement required by an application decides the decomposition strategy in quad trees and usually it depends on the information content lying within a quadrant or a sub-quadrant.

This approach partitions the cover image into blocks which are more homogeneous than the image. Initially, four equal sized square blocks are formed and their further subdivisions are done by comparing a predefined threshold against a parameter associated with each block. The associated parameter is a difference between maximum and minimum pixel values within a block. A block with difference greater than the threshold promotes its subdivision. The threshold value lies in the range of 0 to 1 and is user defined. In addition, minimum and maximum block sizes can also be specified. All these parameters play an important role in deciding the capacity of a cover image. Usually block sizes are defined as power of two, i.e. for a 256×256 image valid block sizes can be taken as 8×8 (minimum), 16×16, 32×32, 64×64, and 128×128 (maximum). Similarly, for a 512×512 image valid block sizes can be taken as 8×8 (minimum), 16×16, 32×32, 64×64, 128×128, and 256×256 (maximum).

3.1.2 Discrete cosine transform (DCT)

Discrete cosine transform (DCT) expresses data as sum of cosine functions, i.e. it transforms signal from spatial to frequency domain. Several areas of science and engineering use concepts of DCT in applications like image compression, audio compression, solving partial differential equations, etc. In comparison to sine functions, fewer number of cosine functions can easily approximate the original signal and hence cosine functions are commonly utilized in compression exertions.

The 2-dimensional DCT transforms $N \times N$ block of a gray scale image f using Eq. 3.1.

$$F(u,v) = C_u C_v \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) cos\left[\frac{(2x+1)u\pi}{2N}\right] cos\left[\frac{(2y+1)v\pi}{2N}\right] \tag{3.1}$$

where C_{u} and C_{v} are normalization coefficients as given in Eq. 3.2, and $f(x,y)$ denotes a pixel located at coordinates (x,y) in f.

$$C_u, C_v = \begin{cases} \sqrt{1/N}, & \text{if } u,v = 0 \\ 1, & \text{otherwise.} \end{cases} \tag{3.2}$$

Similarly, an inverse DCT is obtained using Eq. 3.3.

$$f(x,y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C_u C_v F(u,v) cos\left[\frac{(2x+1)u\pi}{2N}\right] cos\left[\frac{(2y+1)v\pi}{2N}\right] \tag{3.3}$$

DCT possesses strong energy compaction property and most of the energy is compacted in low frequency components that actually represent the correlated image information. This fact is utilized in JPEG image compression which discards high frequency components during quantization. In the domain of steganography, these high frequency components can be used to embed data without affecting the image quality adversely.

3.2 General framework

Fig. 3.1 shows a general framework of the presented steganography approach to embed secret text message within a cover image. The approach is block based, thus the first step is to divide the chosen cover image into square blocks. In this work the concept of quad trees are utilized to get divisions of irregular sizes. Cover image partitioning is followed by the computation of DCT. After these two essential steps the secret text message is processed and embedded in blocks of the cover image. The resulting stego image (cover image enclosing secret message) and a stego key are then transmitted over the communication channel, thus completing the embedding process. For discovering the secret text message from the stego

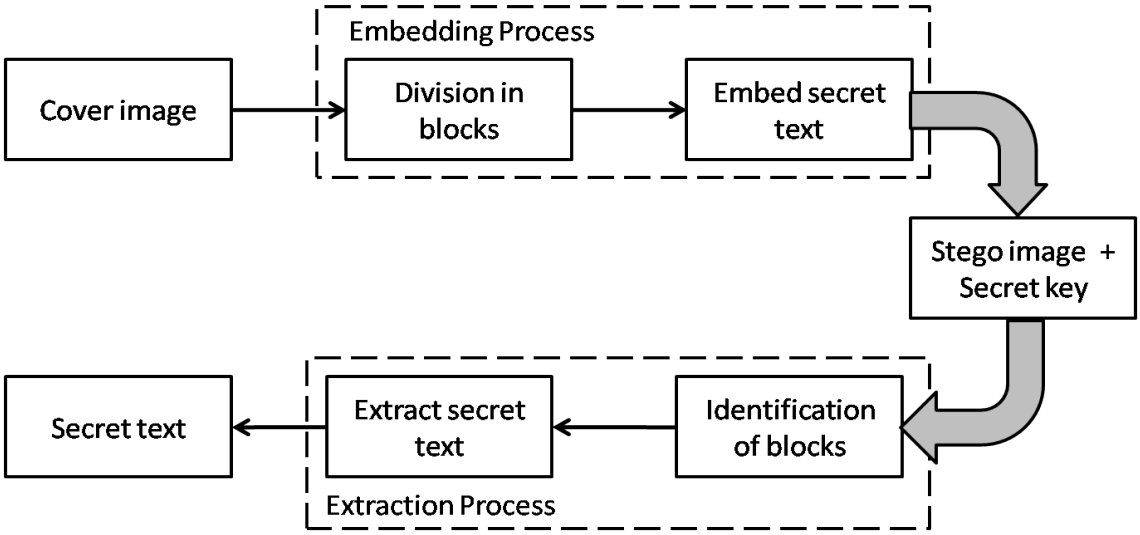


Figure 3.1: General framework of a presented block based transform domain steganographic system.

image, extraction process uses the block information included in the stego key. Once the message is taken out, the extraction process ends.

3.3 Embedding process

Fig. 3.2 demonstrates the complete insight of the embedding process. The detailed steps, to be applied individually on the three channels (R, G, B) of a cover image I, are explained in the following text. Let, C represents a cover image in any one color channel at any moment of time. BS_{min} and BS_{max} represent the minimum block size and maximum block size, respectively.

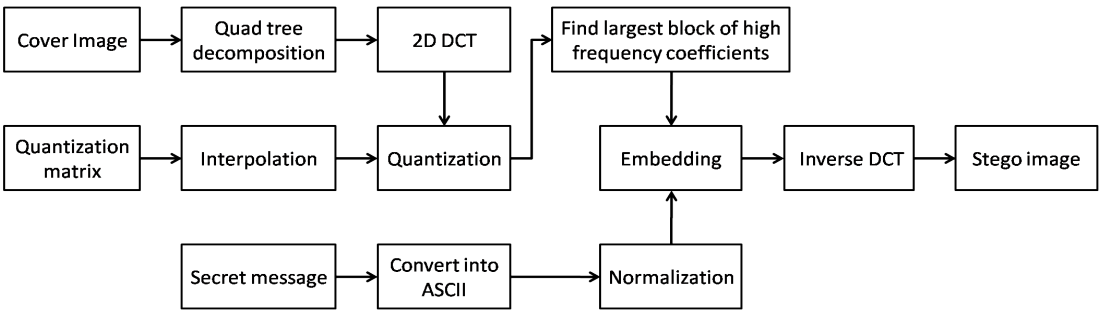


Figure 3.2: A block diagram showing the complete embedding process.

Step 1: Choose a threshold. Decide BS_{min} and BS_{max}. Use these parameters for quad tree decomposition of C to obtain N quad tree blocks with sizes $k \times k$ as shown in Fig. 3.3. k will take any value in between BS_{min} and BS_{max}.

$$C_{qtd} = qtdecomp(C) \tag{3.4}$$

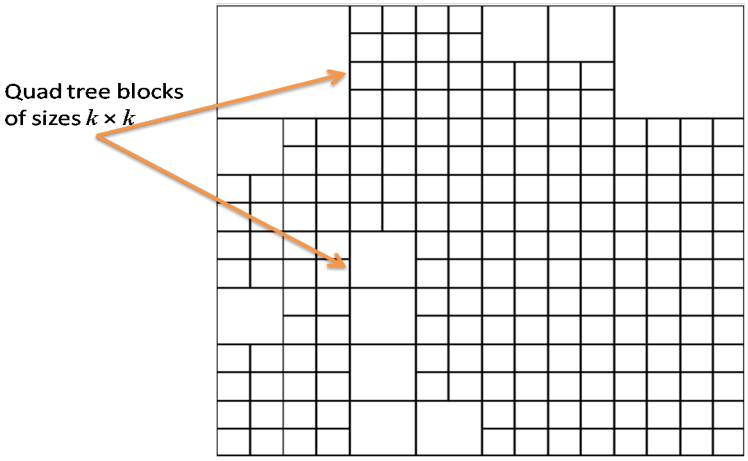


Figure 3.3: A cover image partitioned into N quad tree blocks of sizes $k \times k$.

Step 2: For each of the N quad tree blocks, find 2D-DCT. That is, for each $C_{\text{qtd,sk}}$ find D_{sk}. Here $C_{\text{qtd,sk}}$ is the s^{th} quad tree block in individual blocks of size $k \times k$ and D_{sk} is its 2D-DCT.

$$D = DCT(C_{qtd}) \tag{3.5}$$

Step 3: Resize the quantization matrix shown in Fig. 3.4 according to the quad tree square block with side k using interpolation as in Eq. 3.6.

$$Q' = resize(Q, k \times k) \tag{3.6}$$

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Figure 3.4: The standard 8×8 quantization matrix.

Step 4: Quantize each D_{sk} by dividing elements within a block with a resized quantization matrix obtained in Eq. 3.6. Here D_{sk} is 2D-DCT of the s^{th} quad tree block in individual blocks of size $k \times k$. Quantization intact low energy coefficients, thus helps in easy reconstruction of an image. In addition it facilitates replacement of less important DCT coefficients.

$$Q_{dct} = D/Q' \tag{3.7}$$

Step 5: Within each quad block in Q_{dct}, find the largest block of high frequency coefficients as shown in Fig. 3.5. Let $b_{sk} \times b_{sk}$ denotes the size of s^{th} adaptive block in individual blocks of size $k \times k$, the maximum number of characters that can be embedded within a single color channel (say i) of a given cover image can be computed as in Eq. 3.8. Sum T_i, $i \in$ R, G, and B, values for

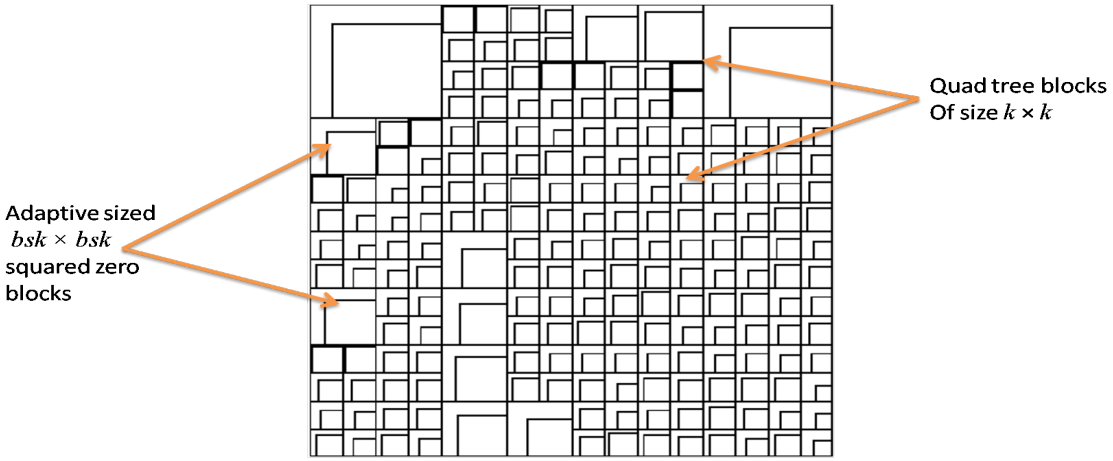


Figure 3.5: A quad tree partitioned cover image marked with adaptive blocks of size $b_{sk} \times b_{sk}$.

all color channels of a cover image to get total capacity as given in Eq. 3.9

$$T_i = \sum_k \sum_s b_{sk} \times b_{sk} \tag{3.8}$$

$$T_{total} = \sum_i \sum_k \sum_s b_{sk} \times b_{sk} \tag{3.9}$$

Step 6: Normalize A_{t}, ASCII values of text, using Eq. 3.10. Normalization is a necessary step as it unifies ASCII values of text with the DCT coefficients obtained for the cover image.

$$A'_t = A_t/127 \tag{3.10}$$

Step 7: Replace each s^{th} adaptive block of high frequency DCT coefficients in individual blocks of size $k \times k$ with the normalized text values, A'_{t}. This will embed text in a single channel i of a cover image. If all N blocks are utilized this implies that the i^{th} color channel of the cover image is embedded to its maximum capacity. Let D_k embedded with text in any single color channel be denoted by $Q_{dctTEmd}$.

Step 8: Compute inverse DCT of all the N blocks within $Q_{dctTEmd}$ using Eq. 3.11.

$$S = idct(Q_{dctTEmd}) \tag{3.11}$$

Following the discussed procedure S images are obtained for the three color channels (Red, Green, and Blue) and combined to create the stego image for a given co-

ver image. Additionally, the stego key which is needed for extraction is generated. The key contains information about the quad tree blocks and the corresponding adaptive blocks of high frequency coefficients with respect to each of the three color channels. It also maintains the order of blocks which is followed to embed the secret text.

3.4 Extraction process

The extraction process is simpler as compared to a complex embedding process and is shown in Fig. 3.6. Extracting a secret text from the stego image uses the secret key to determine the size and location of quad tree blocks and their respective adaptive blocks of high frequency coefficients. For each quad tree block, 2D-DCT is computed using Eq. 3.5 and the values from the corresponding adaptive blocks are extracted. Then the extracted values are concatenated to get a normalized text, A'_t. ASCII values of the original text recovered using Eq. 3.12 are mapped to get the secret text.

$$A_t = 127 \times A'_t \tag{3.12}$$

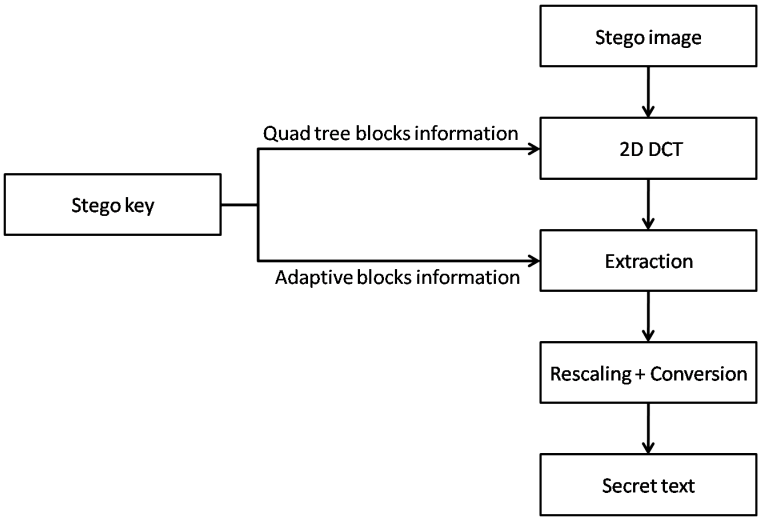


Figure 3.6: A block diagram showing the complete extraction process.

3.5 Illustrative examples

Working of the presented block based transform domain steganography approach is explained using cover images of sizes 256×256 and 512×512. In both cases

the cover images are embedded to their maximum available capacities.

3.5.1 Using 256×256 cover image

The step by step process to embed secret text in a cover image of size 256×256 is shown in Fig. 3.7. Firstly, the original *House* image is divided into variable sized but highly coherent blocks using quad tree decomposition. In this example, threshold is set to 0.84, minimum block size (BS_{min}) to 32×32, and maximum block size (BS_{max}) to 128×128. Keeping maximum block size to 128×128 for a cover image of size 256×256 splits the original image into four blocks without checking the required threshold criteria. Further increment of BS_{max} may resemble a without block based steganographic approach if predefined threshold criteria does not meet.

In the considered example, 10 blocks are formed: three 128×128 blocks, three 64×64 blocks, and four 32×32 blocks. Then 2D DCT is performed on each of the obtained blocks, which transforms the cover image into frequency domain. The transformed blocks are quantized by dividing each of the 10 blocks by a quantization matrix, interpolated to the size of a block. Then, within each quantized block, the largest block of high frequency coefficients i.e. zero valued block is found in the lower right corner as shown in Fig. 3.7. Finally, the obtained zero

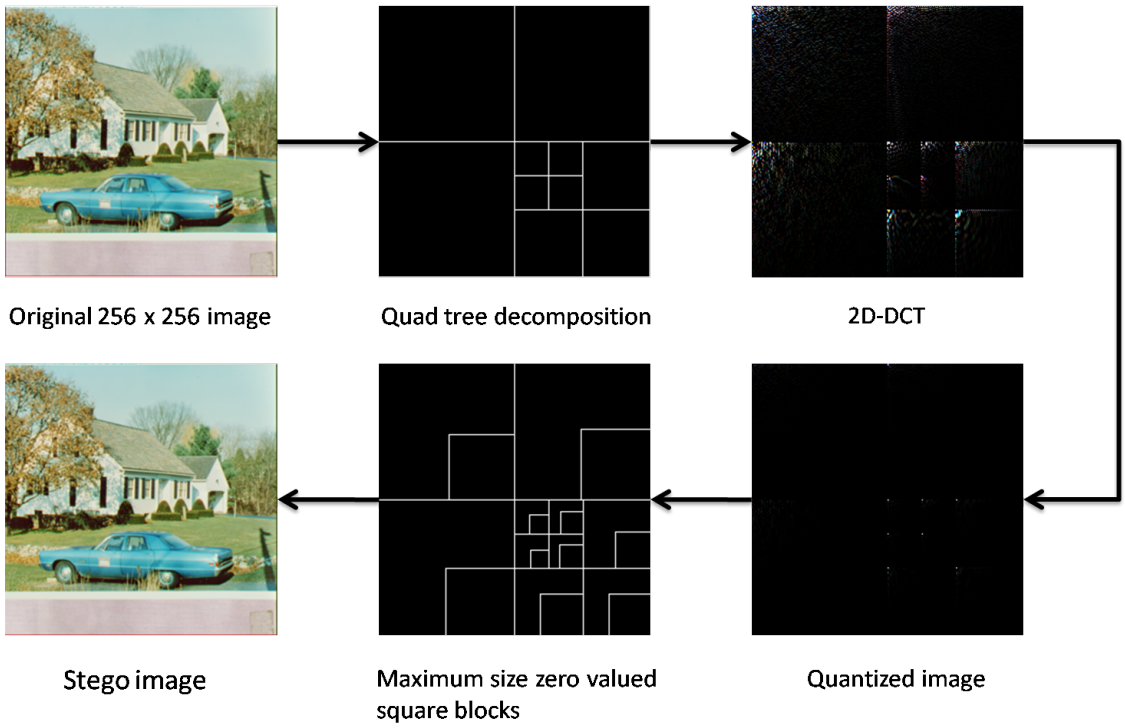


Figure 3.7: Embedding secret text in a 256×256 cover image at threshold of 0.84, minimum block size of 32×32 and maximum block size of 128×128.

valued square blocks in the transformed and quantized DCT image are replaced with normalized ASCII values of a secret text. The complete capacity of *House* cover image is utilized by embedding 55,534 characters in this example. After replacement of high frequency coefficients inverse DCT is computed to get the required stego image. Visual examination of original and stego images clearly states the difficulty involved in differentiating them.

In addition to stego image, a stego key is generated and transmitted to facilitate extraction. The information contained in the stego key for the *House* image is shown in Table 3.1. The positions of DCT blocks are same in any of the three color channels, but the locations of adaptive high frequency DCT coefficients blocks vary from one color channel to another. All the blocks have square shape and are thus identified by two pairs of pixel positions: one is the upper left pixel positions pair and another is the lower right. Moreover, lower right pixel positions pair of DCT blocks and adaptive blocks in any color channel coincide with each other. Here pixel position is taken in the form of a (row, column) pair. (1,1) and (256,256) signify the upper left and lower right corners of a cover image. The order of blocks followed during embedding remains the same in the presented approach irrespective of the color channel. The blocks are sorted in decreasing order of sizes and same sized blocks follow column wise arrangement. Fig. 3.8 clarifies all these statements and considering them, following is the description of each component constituting the stego key.

- Order: Order of blocks (top to bottom and left to right).
- B_{ur}: Row number of upper left pixel position of a DCT block.

Table 3.1: Stego key obtained for a 256×256 stego image in Fig. 3.7.

Order	DCT blocks				Adaptive Blocks					
	B_{ur}	B_{uc}	B_{lr}	B_{lc}	R Channel		G Channel		B channel	
					A_{ur}	A_{uc}	A_{ur}	A_{uc}	A_{ur}	A_{uc}
1	1	1	128	128	68	68	68	68	68	68
2	129	1	256	128	193	65	193	65	185	57
3	1	129	128	256	63	191	63	191	63	191
4	193	129	256	192	217	153	218	154	216	152
5	129	193	192	256	159	223	160	224	155	219
6	193	193	256	256	217	217	222	222	216	216
7	129	129	160	160	143	143	142	142	141	141
8	161	129	192	160	176	144	175	143	171	139
9	129	161	160	192	140	172	143	175	140	172
10	161	161	192	192	171	171	170	170	170	170

- B_{uc}: Column number of upper left pixel position of a DCT block.
- B_{lr}: Row number of lower right pixel position of a DCT block.
- B_{lc}: Column number of lower right pixel position of a DCT block.
- A_{ur}: Row number of upper left pixel position of an adaptive block.
- A_{uc}: Column number of upper left pixel position of an adaptive block.

Stego key contains data for each block in the stego image. In the considered example there are 10 blocks and hence 10 rows. Size of blocks with order 1, 2, and 3 is 128×128. Those with order 4, 5, and 6 have size 64×64. Remaining blocks are of 32×32 size. From Table 3.1, it can be seen that the first block used for embedding has upper left pixel position as (1,1) and lower right pixel position as (128,128). For this DCT block, the position of adaptive blocks in three color channels is same as is shown by the alike upper left pixel position value, i.e.

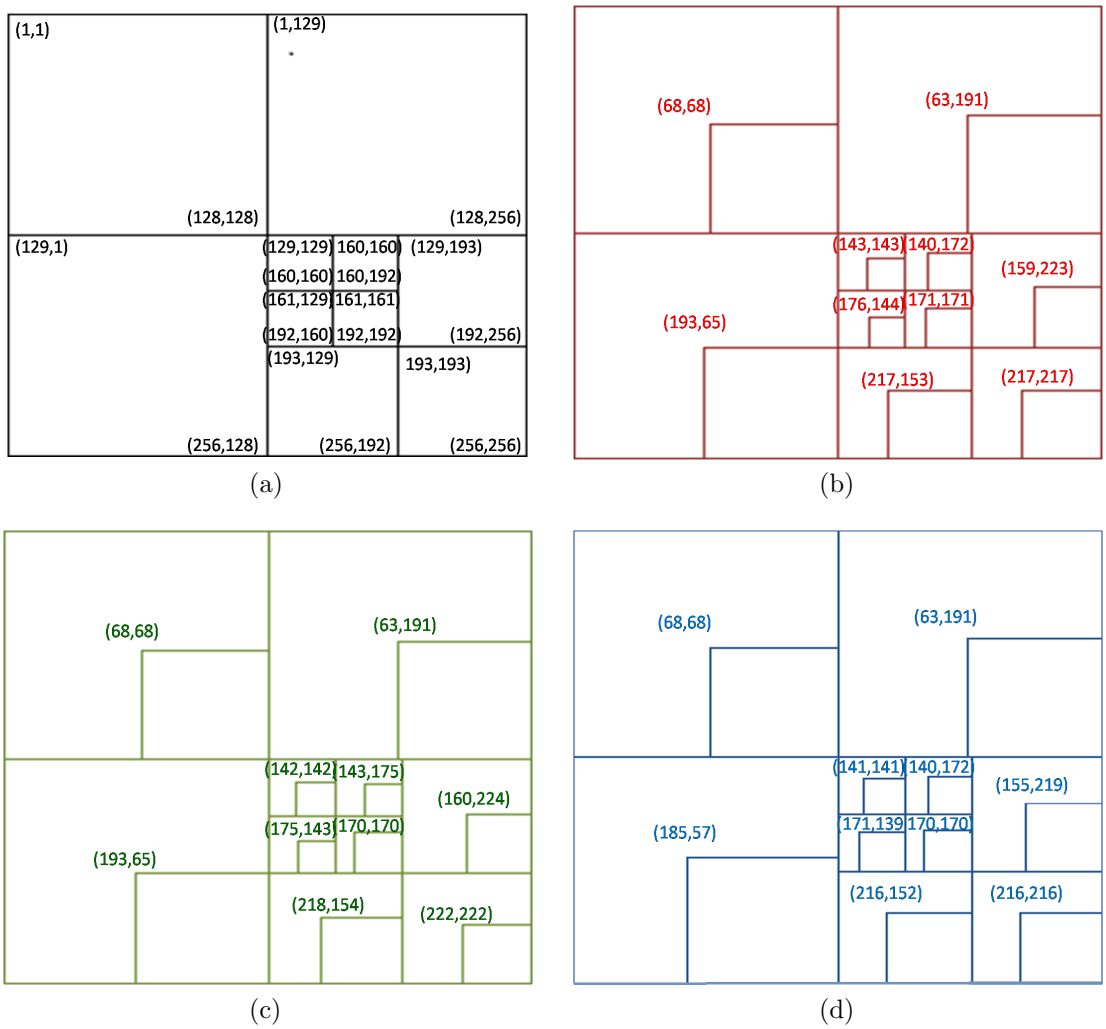


Figure 3.8: (a) DCT blocks. (b) R channel adaptive blocks. (c) G channel adaptive blocks. (d) B channel adaptive blocks.

(68,68). On the other hand, for block with order 6 the upper left and lower right pixel positions are (193,193) and (256,256), respectively. Moreover, the positions of adaptive blocks are different in all the color channels in this case as is revealed from dissimilar (A_{ur}, A_{uc}) pair values in R, G, and B color channels.

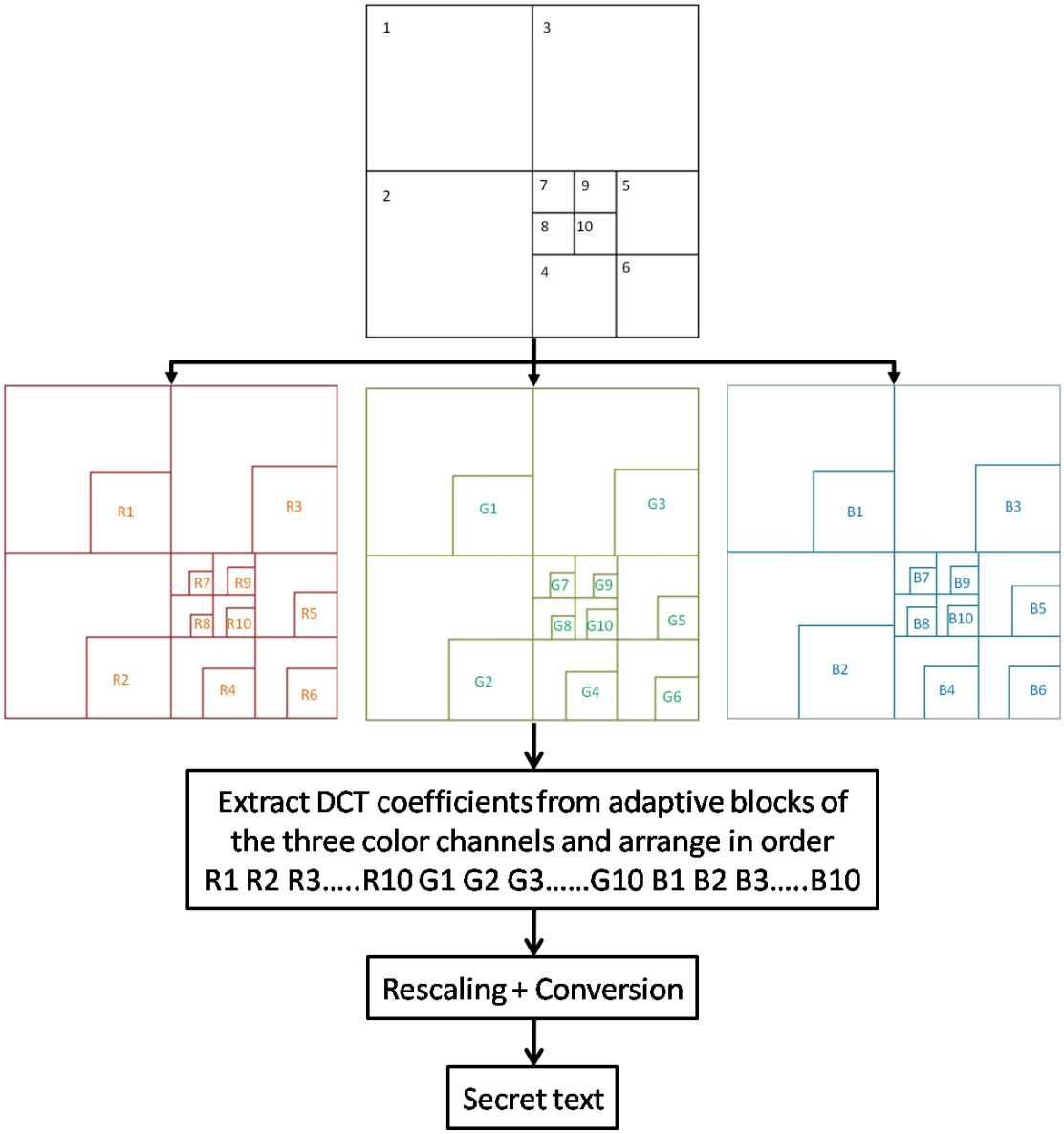


Figure 3.9: Extraction of secret text from 256 × 256 stego image.

The step by step execution of the extraction process at the receiver side is shown in Fig. 3.9. It starts by dividing the stego image into variable blocks using the information contained in columns with heads B_{ur}, B_{uc}, B_{lr}, and B_{lc}. 2D DCT of each block is computed for all the three color channels. The coefficients lying in the corresponding adaptive regions of the three color channels are extracted and arranged following the order in which information is found in the stego key. Red channel obtains adaptive blocks using information contained in columns with heads R Channel, B_{lr}, and B_{lc}. Similarly, Green and Blue channels use information

contained in columns with heads G Channel and B Channel, respectively in place of R Channel. Rest of the two columns, B_{lr} and B_{lc}, remain same. Finally, extracted coefficients are re-scaled and converted into characters. This reveals the secret text and thus ends the extraction process.

3.5.2 Using 512×512 cover image

The process for embedding secret text in a cover image of size 512×512 is detailed in Fig. 3.10. Firstly, quad tree decomposition divides the original *Building* image into variable sized and highly coherent blocks. In this example, threshold is set to 0.73, minimum block size (BS_{min}) to 64×64, and maximum block size (BS_{max}) to 256×256. Keeping maximum block size to 256×256 for a cover image of size 512×512 splits the original image into four blocks without checking the required threshold criteria. Increasing BS_{max} to the next level may work similar to a without block based steganographic approach if predefined threshold criteria does not meet.

In the considered example, 22 blocks are formed: one 256×256 block, nine 128×128 blocks, and twelve 64×64 blocks. 2D DCT of all the 22 blocks is obtained that transforms the cover image into frequency domain. Each of the transformed blocks is quantized by division through a quantization matrix which is first interpolated

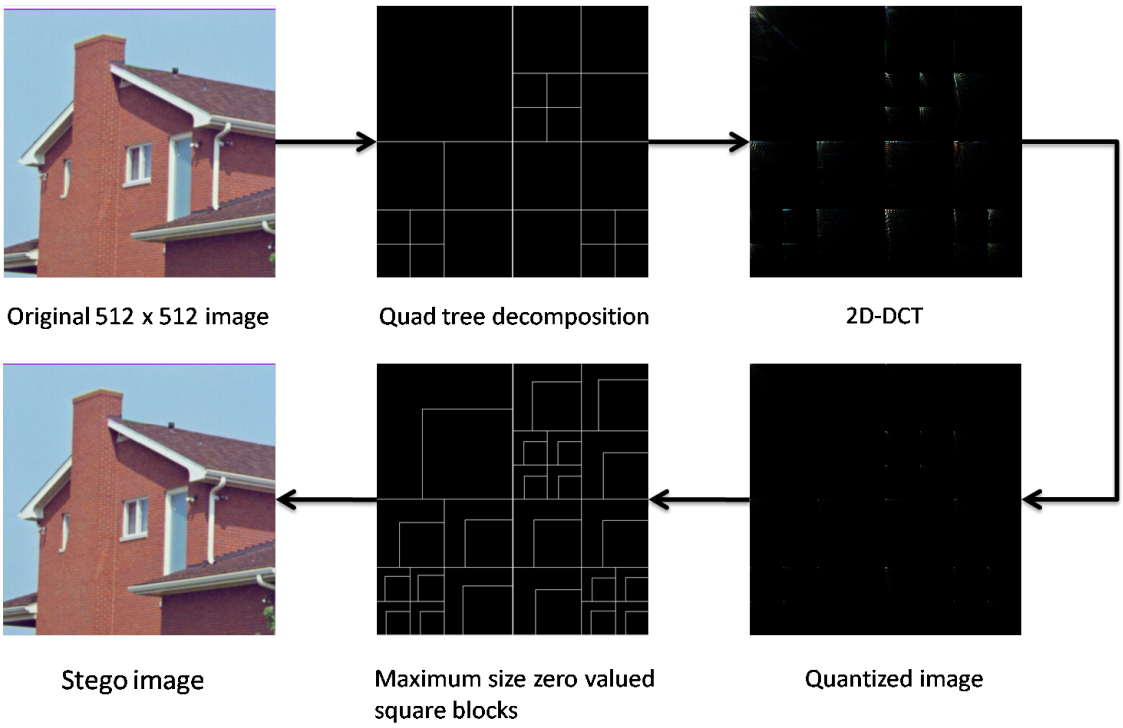


Figure 3.10: Embedding secret text in a 512×512 cover image at threshold of 0.73, minimum block size of 64×64 and maximum block size of 256×256.

to the size of a block. The largest block of high frequency coefficients i.e. zero valued block is then found in the lower right corner within each quantized block as shown in Fig. 3.10. Finally, the obtained blocks of high frequency coefficients in the transformed and quantized DCT image are replaced with normalized ASCII values of a secret text. The complete capacity of *Building* cover image is utilized by embedding 3,72,332 characters in this example. After replacement of high frequency coefficients inverse DCT is computed to get the required stego image. For this cover image as well it's difficult to differentiate between original and stego images.

Similar to the previous example, in addition to stego image, a stego key is generated and transmitted to facilitate extraction. The information contained in the stego key for the *Building* image is shown in Table 3.2. The statements related to order, DCT blocks, and adaptive blocks discussed earlier hold here as well. In this example, (512,512) signifies the lower right corner of a cover image. Fig. 3.11 clarifies all the statements in the context of a 512×512 cover image. Stego key

Table 3.2: Stego key obtained for a 512×512 stego image in Fig. 3.10.

Order	DCT blocks				Adaptive Blocks					
	B_{ur}	B_{uc}	B_{lr}	B_{lc}	R Channel		G Channel		B channel	
					A_{ur}	A_{uc}	A_{ur}	A_{uc}	A_{ur}	A_{uc}
1	1	1	256	256	88	88	89	89	96	96
2	257	1	384	128	301	45	301	45	305	49
3	257	129	384	256	296	168	296	168	299	171
4	385	129	512	256	420	164	420	164	429	173
5	1	257	128	384	37	293	31	287	34	290
6	257	257	384	384	296	296	295	295	302	302
7	385	257	512	384	427	299	428	300	429	301
8	1	385	128	512	33	417	24	408	10	394
9	129	385	256	512	170	426	167	423	174	430
10	257	385	384	512	297	425	299	427	305	433
11	385	1	448	64	402	18	398	14	399	15
12	449	1	512	64	468	20	467	19	471	23
13	385	65	448	128	400	80	405	85	407	87
14	449	65	512	128	468	84	469	85	470	86
15	129	257	192	320	149	277	147	275	152	280
16	193	257	256	320	214	278	211	275	216	280
17	129	321	192	384	149	341	147	339	151	343
18	193	321	256	384	214	342	210	338	216	344
19	385	385	448	448	405	405	402	402	405	405
20	449	385	512	448	467	403	467	403	470	406
21	385	449	448	512	404	468	406	470	407	471
22	449	449	512	512	468	468	469	469	474	474

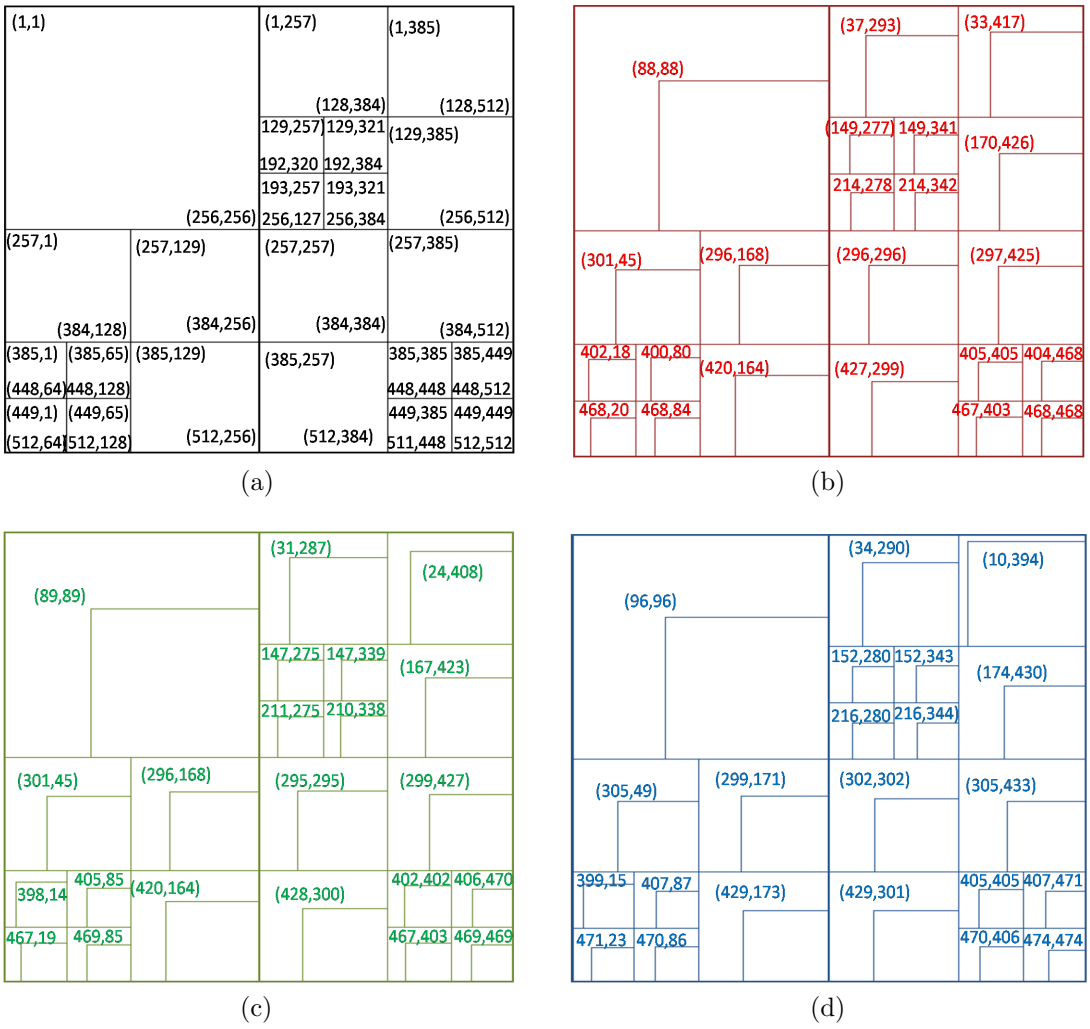


Figure 3.11: (a) DCT blocks. (b) R channel adaptive blocks. (c) G channel adaptive blocks. (d) B channel adaptive blocks.

contains data for each of the 22 blocks. Size of block with order 1 is 256×256. Those with order 2 to 10 have size 128×128. Remaining blocks are of 64×64 size. From Table 3.2, it can be seen that the first block used for embedding has upper left pixel position as (1,1) and lower right pixel position as (256,256). For this DCT block, the positions of adaptive blocks are different in all the color channels as is revealed from dissimilar (A_{ur}, A_{uc}) pair values in R, G, and B color channels. The adaptive block's upper left corner is at (88,88) for R channel, (89,89) for G channel, and (96,96) for B channel.

At the receiver side, extraction process uses the information contained in the stego key. It starts by dividing the stego image into variable blocks, then computing their 2D DCT in all the three color channels, extracting coefficients from the adaptive regions, and arranging them following the order information. Finally, extracted coefficients are re-scaled and converted into characters. This reveals the

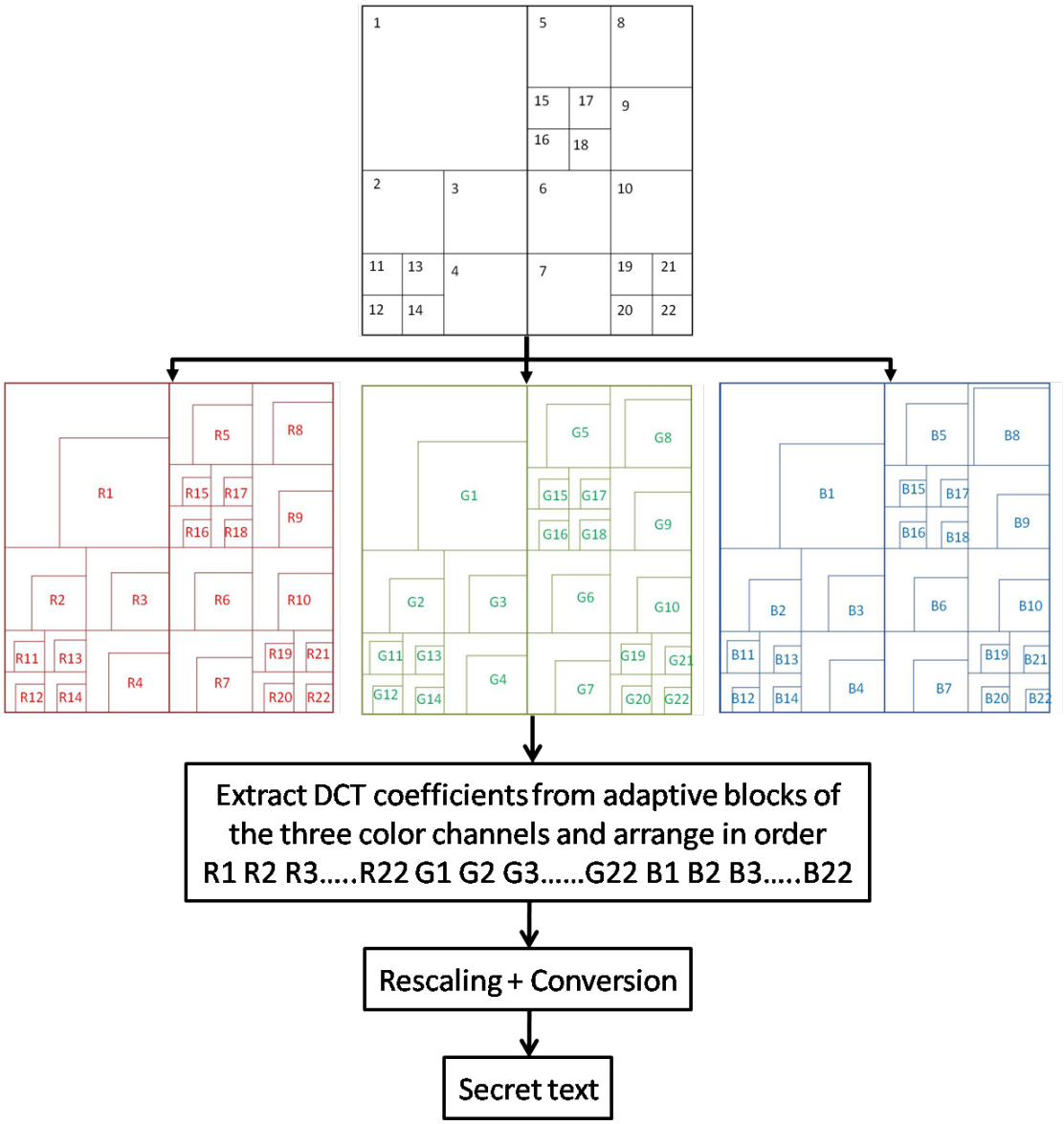


Figure 3.12: Extraction of secret text from 512×512 stego image.

secret text and thus ends the extraction process. The step by step execution of this process is shown in Fig. 3.12

3.6 Summary

This chapter details the block based transform domain steganography approach presented in this work. The in depth explanation of two phases of a steganographic system, i.e. embedding and extraction phases, is followed by two illustrative examples. In both cases, the employed cover images (of sizes 256×256 and 512×512) are embedded to their maximum available capacity. Although no quantitative evaluation parameter is discussed in this chapter, but the visual examination clearly

shows efficacy of the discussed steganography approach. It is evident from Fig. 3.7 and Fig. 3.10 that differentiating an original image from a stego image using naked eyes is truly complex.

Chapter 4

Results and Discussions

The block based transform domain steganography approach presented in this work is evaluated using some well-known parameters, i.e. PSNR and SSIM, in addition to capacity. Results are generated by varying values of minimum block sizes (BS_{min}) within a certain range. The obtained results prove the effectiveness of the presented scheme over the existing spatial and transform domain approaches (without block and fixed block based) in terms of all the considered parameters.

The chapter starts with a description of experimental setup in Section 4.1 followed by definitions of employed performance parameters in Section 4.2. Section 4.3 discusses the results with respect to the maximum capacity values reported in all the considered cases. Lastly, Section 4.4 compares the results obtained by the presented approach with state-of-the-art spatial and transform domain steganographic approaches.

4.1 Experimental setup

The presented steganographic approach is implemented in MATLAB R2014a. All experiments are performed on a PC with Intel Core 2 Quad 2.60 GHz processor, 4GB RAM, 500GB HDD, and Windows 7 64-bit operating system. The applicability of the presented method is evaluated using a public database USC-SIPI-ID [41]. It contains standard images of *Baboon*, *Building*, *F16jet*, *House*, *Lena* and *Trees* in sizes 256×256 and 512×512 as shown in Fig. 4.1. All the images are exhaustively tested on performance parameters for several combinations of threshold, BS_{min}, and maximum block sizes (BS_{max}).

During experimentations, different threshold values and minimum block sizes are used for dividing the cover image in quad tree blocks, resulting in different capacity, PSNR and SSIM values. The threshold values for quad tree decomposition can range from 0 to 1. For a 256×256 sized cover image, minimum block size (BS_{min}) in varied from 8×8 to 128×128 (BS_{max}). Similarly, for a cover image with size

(a) (b) (c)

(d) (e) (f)

Figure 4.1 USC-SIPI-ID database images used as cover images during experiments. (a) Baboon. (b) Building. (c) F16 jet. (d) House. (e) Lena. (f) Trees.

512×512, BS_{max} is set to 256×256 and BS_{min} starts from 8×8. These values decide the structure of resultant quad tree depending on the amount of correlation in the image.

4.2 Performance parameters

The quality of the stego image is measured using peak signal to noise ratio (PSNR) and structural similarity index (SSIM). The most important test for steganography is the human perceptibility, which is quite subjective. Mean squared error (MSE) is popularly used to objectively assess an image quality and is given by Eq. 4.1.

$$MSE = \frac{1}{mn} \sum_{0}^{m-1} \sum_{0}^{n-1} \|f(i,j) - g(i,j)\|^2 \tag{4.1}$$

where f is the original image and g is the stego image, both of size $m \times n$.

PSNR uses MSE and is the ratio of maximum possible power of signal to the power of corrupting noise. Its unit is decibels and is defined as in Eq. 4.2. Higher PSNR value signifies better image quality.

$$PSNR = 20log_{10}\left(\frac{L-1}{\sqrt{MSE}}\right) \tag{4.2}$$

where L is number of gray levels in image.

However, MSE is dependent on image intensity and scaling, thus is sensitive to minor pixel variations. A better image assessment measure is SSIM, as human eye can easily extract structural information and hence signal structure plays an important role during assessment [13]. SSIM is based on this fact and is defined as in Eq. 4.3. SSIM value close to 1 represents better result.

$$SSIM = \frac{(2\mu_f\mu_g + C_1)(2\sigma_{fg} + C_2)}{(\mu_f^2 + \mu_g^2 + C_1)(\sigma_f^2 + \sigma_g^2 + C_2)} \tag{4.3}$$

where f is the original image and g is the stego image. μ_f, μ_g are means and σ_f^2, σ_g^2 are variances of f and g respectively. σ_{fg} is the covariance of images f and g. C_1 and C_2 are stabilization coefficients defined as $C_1 = (k_1 L)^2$ and $C_2 = (k_2 L)^2$, where $L = 2^{(\text{Number of bits per pixel})}$ - 1. Default values of k_1 and k_2 are 0.01 and 0.03 respectively.

4.3 Results for maximum capacity

A cover image when provided with different values of threshold, minimum block size (BS_{min}), and maximum block size (BS_{max}) gives different sizes of quad tree blocks. Such variability when propagated further result in diverse sizes of adaptive zero coefficients blocks. Thus, capacities in most of the cases are distinctive. And when text is embedded to the entire capacity of the cover image, dissimilar stego images are obtained with varying PSNR and SSIM values.

Comparing stego images obtained with different combinations of threshold, BS_{min}, and BS_{max} for a 256×256 cover image. Fig. 4.2 shows original *Lena* image and compares stego images obtained using different constraints settings. Visually all images are looking similar but observing quality assessment measures, i.e. PSNR and SSIM along with capacity then variations are clear. BS_{max} is fixed to 128×128 in all the cases. When BS_{min} is 8×8, the maximum capacity achieved is 1,12,408 characters at a threshold of 0.05. This combination results in a PSNR of 36.4737 and SSIM of 0.9947. Changing BS_{min} to 16×16, alters achievable maximum capacity to 1,01,729, which is obtained at threshold of 0.03. Also, the reported PSNR and SSIM parameters are 38.6219 and 0.9967, respectively. Similar variable observations are made for other possible combinations as is shown in Fig. 4.2.

Comparing stego images obtained with different combinations of threshold, BS_{min}, and BS_{max} for a 512×512 cover image. The original *Baboon*

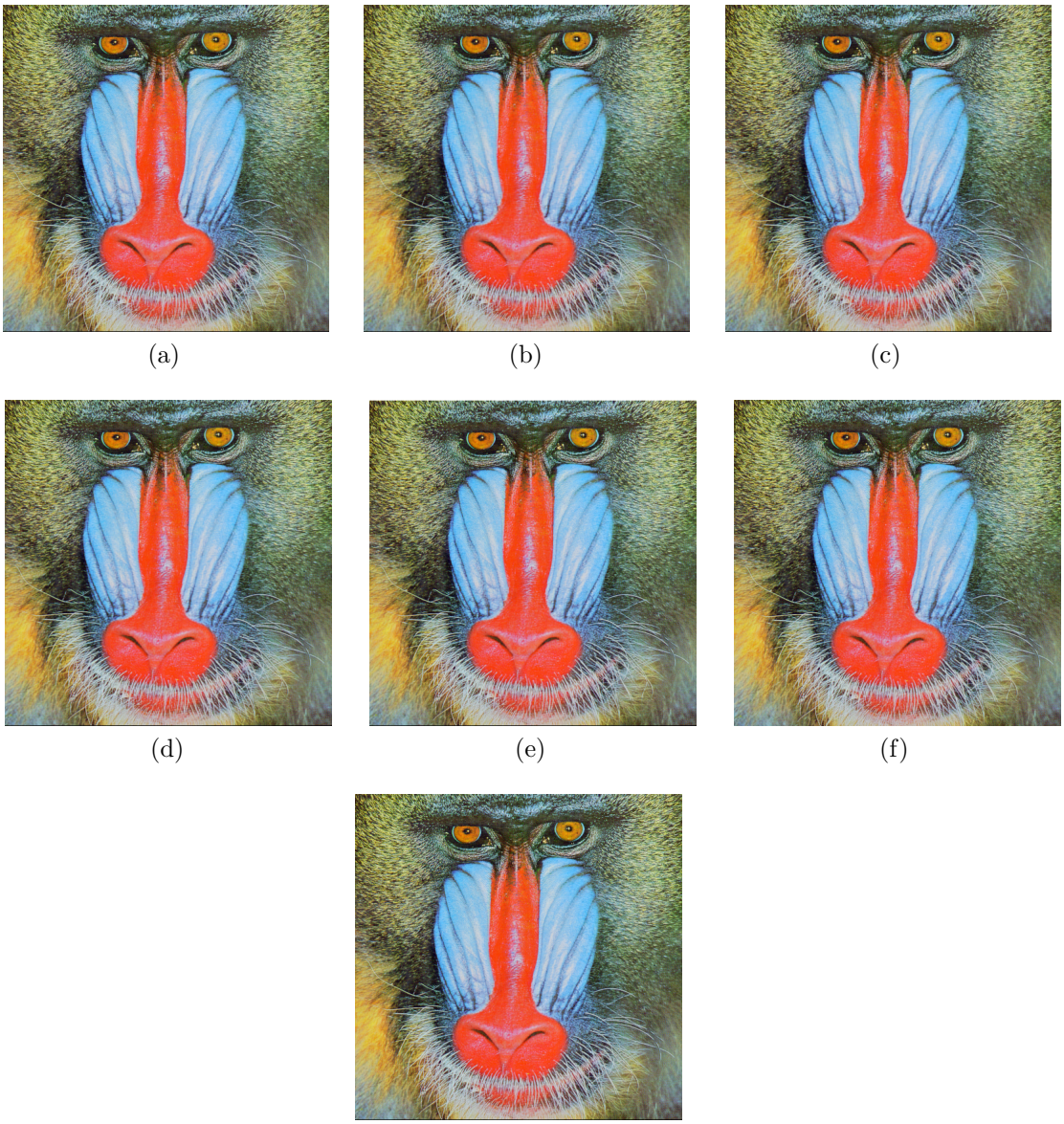
(a) (b) (c)

(d) (e) (f)

Figure 4.2 Original *Lena* image and stego images for different constraint settings embedded to their maximum capacities. th = threshold and c = capacity. (a) Original 256×256 image. (b) $th = 0.05$, $BS_{min} = 8 \times 8$, c = 1,12,408, PSNR = 36.4737, SSIM = 0.9947. (c) $th = 0.03$, $BS_{min} = 16 \times 16$, c = 1,01,729, PSNR = 38.6219, SSIM = 0.9967. (d) $th = 0.1$, $BS_{min} = 32 \times 32$, c = 87,243, PSNR = 39.3695, SSIM = 0.9972. (e) $th = 0.1$, $BS_{min} = 64 \times 64$, c = 70,611, PSNR = 40.4086, SSIM = 0.9978. (f) $th = 0.1$, $BS_{min} = 128 \times 128$, c = 57,455, PSNR = 41.5034, SSIM = 0.9982.

image and various resultant stego images for different combinations are shown in Fig. 4.3. Here as well the visual differentiation is difficult but quantitative analysis in terms of capacity, PSNR, and SSIM clarifies everything. Setting BS_{min} to 8×8 achieves a maximum capacity of 3,06,251 characters at a threshold of 0.02 with PSNR of 30.7401 and SSIM of 0.9672. Increasing BS_{min} to 16×16 and keeping threshold as 0.01 reduces the maximum achievable capacity to 2,56,505 characters. The resultant stego image reports PSNR as 32.3282 and SSIM as 0.9763. Here as well, similar variations can be observed for other possible combinations as is shown in Fig. 4.3.

From both these examples, it is evident that as the BS_{min} increases, maximum capacity decreases that consequently increases the PSNR and SSIM values. In other words increasing minimum block size is responsible for an improved quality of stego image. This is because as the minimum block size increases, the amount of correlation in each block decreases. Thus, more coefficients of a block are required to represent the image. Hence, the size of maximum zero valued block



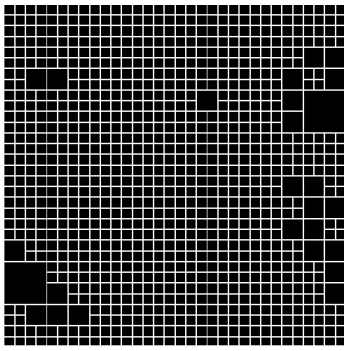
(a) (b) (c)

(d) (e) (f)

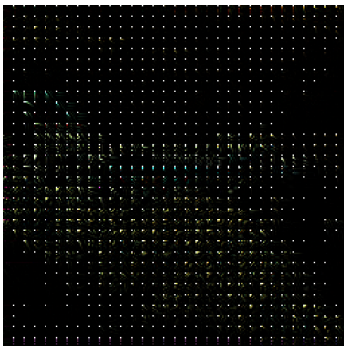
(g)

Figure 4.3 Original *Baboon* image and stego images for different constraint settings embedded to their maximum capacities. th = threshold and c = capacity. (a) Original 512×512 image. (b) $th = 0.02$, $BS_{min} = 8 \times 8$, c = 3,06,251, PSNR = 30.7401, SSIM = 0.9672. (c) $th = 0.01$, $BS_{min} = 16 \times 16$, c = 2,56,505, PSNR = 32.3282, SSIM = 0.9763. (d) $th = 0.1$, $BS_{min} = 32 \times 32$, c = 2,02,065, PSNR = 33.9103, SSIM = 0.9837. (e) $th = 0.1$, $BS_{min} = 64 \times 64$, c = 1,31,154, PSNR = 36.2376, SSIM = 0.9837. (f) $th = 0.1$, $BS_{min} = 128 \times 128$, c = 89,811, PSNR = 38.6675, SSIM = 0.9955. (g) $th = 0.1$, $BS_{min} = 256 \times 256$, c = 57,854, PSNR = 40.457, SSIM = 0.9966.

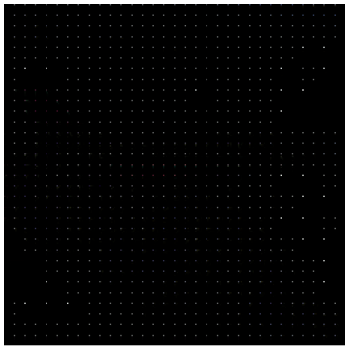
(a) Original image



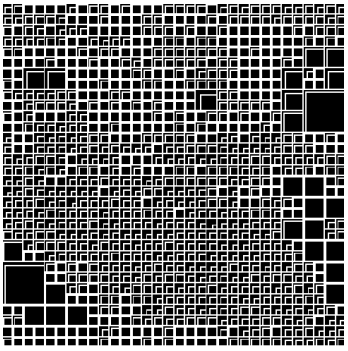
(b) Quad tree decomposition



(c) 2D DCT computation



(d) Quantization



(e) Adaptive block computation

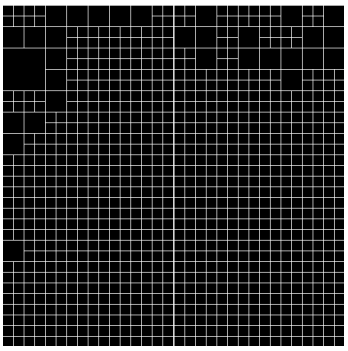
(f) Stego image

Figure 4.4 Embedding secret text in a 256×256 *F16 jet* cover image at threshold of 0.07, minimum block size of 8×8 and maximum block size of 128×128. Capacity = 1,15,396 characters, PSNR = 37.1237, and SSIM = 0.9765.

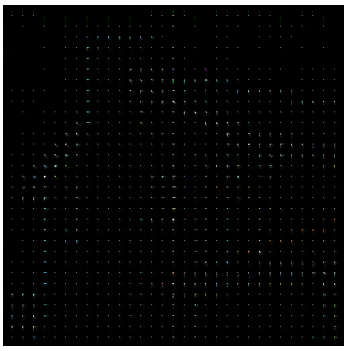
decreases resulting in decreased size of adaptive blocks for embedding text, thus reduced achievable maximum capacity. For additional elucidation, the step by step embedding process as discussed during illustrative examples in Section 3.5 is also shown in this chapter for images *F16 jet* (256×256) and *Building* (512×512). Fig. 4.4 displays intermediate results for *F16 jet* using a threshold of 0.07, BS_{min} of 8×8, and BS_{max} of 128×128. The maximum capacity of 1,15,396 characters is reported for this combination with a PSNR of 37.1237 and SSIM of 0.9765. In a same way, Fig. 4.5 exhibits alike results for *Building* at 0.05 threshold. BS_{min} and BS_{max} are fixed at 16×16 and 256×256, respectively. This combination accounted a maximum capacity of 4,92,086 characters, PSNR of 45.586, and SSIM of 0.9979.

The presented approach is able to achieve high capacity while maintaining desired PSNR and SSIM values. Results are generated by fixing a minimum block size and then threshold value is varied in the range of 0.01 to 0.1 with an increment of 0.01 in each step. Combination that achieves the maximum capacity for image sizes 256×256 and 512×512 are included in Table 4.1 and Table 4.2 respectively. It's clearly visible that the minimum quad tree block size is more dominating

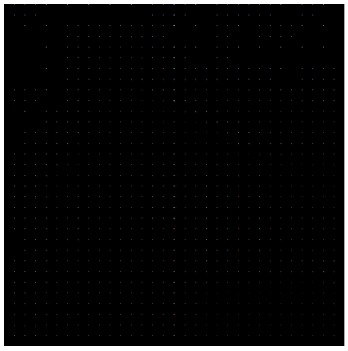
(a) Original image



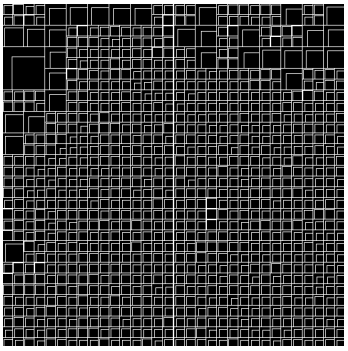
(b) Quad tree decomposition



(c) 2D DCT computation



(d) Quantization



(e) Adaptive block computation

(f) Stego image

Figure 4.5 Embedding secret text in a 512×512 *Building* cover image at threshold of 0.05, minimum block size of 16×16 and maximum block size of 256×256. Capacity = 4,92,086 characters, PSNR = 45.586, and SSIM = 0.9979.

factor as compared to the threshold value in deciding the embedding capacity of a cover image. The minimum block size of 8×8 always yield the maximum capacity among all the other block sizes. Also for a given minimum block size, the capacity remains unchanged for more than one threshold value and the same is specified by a range in Table 4.1 and Table 4.2 . Moreover, the PSNR of above 30 is achieved in all the cases. The SSIM gets reduced a bit, but looking at an increase in the capacity this much increment is considerable. Moreover, to prove imperceptibility of the presented steganography approach, Table 4.3 visually compares each of the original image with the corresponding stego images embedded to their maximum capacities. Clearly, the resulting stego images obtained for both sizes (256×256 and 512×512) look same as that of their original counterparts. Hence, its difficult to judge any existence of a secret text in these images with naked eyes. The reported capacity, PSNR, and SSIM are in the desired range which further establishes acceptance of the presented approach.

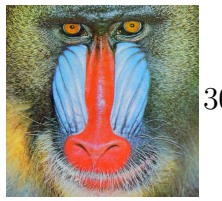
Table 4.1: Maximum capacity, PSNR, and SSIM obtained for a given minimum block size (BS_{min}) and a threshold value using 256×256 images.

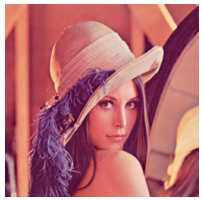
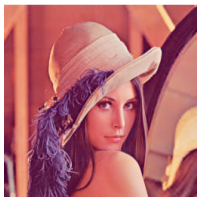
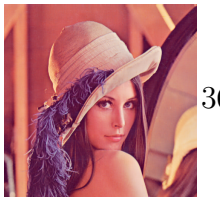
Image	BS_{min}	Threshold	Capacity	PSNR	SSIM
Baboon	8×8	0.01-0.09	78,558	32.0056	0.9706
	16×16	0.01-0.1	66,713	33.6566	0.9803
	32×32	0.01-0.1	55,933	35.0103	0.9857
	64×64	0.01-0.1	48,005	35.8301	0.9877
	128×128	0.01-0.1	43,413	35.9226	0.9885
Building	8×8	0.09	1,25,076	37.7581	0.9900
	16×16	0.01-0.04	1,13,051	39.6168	0.9931
	32×32	0.09-0.1	98,347	40.7161	0.9946
	64×64	0.01-0.1	85,413	41.4086	0.9957
	128×128	0.01-0.1	66,501	42.2478	0.9963
F16jet	8×8	0.07	1,15,396	37.1237	0.9765
	16×16	0.07-0.08	1,04,101	39.1493	0.9822
	32×32	0.01-0.1	91,537	40.0874	0.9851
	64×64	0.01-0.1	71,180	41.8677	0.9912
	128×128	0.01-0.1	56,729	42.0817	0.9939
House	8×8	0.06	1,06,215	35.7482	0.9870
	16×16	0.06	96,783	37.607	0.9908
	32×32	0.01-0.1	84,318	38.8049	0.9931
	64×64	0.01-0.1	67,153	39.7551	0.9953
	128×128	0.01-0.1	53,048	40.7196	0.9964
Lena	8×8	0.09	1,12,449	36.4788	0.9947
	16×16	0.01-0.03	1,01,729	38.6219	0.9967
	32×32	0.01-0.1	87,243	39.3695	0.9972
	64×64	0.01-0.1	70,611	40.4086	0.9978
	128×128	0.01-0.1	57,455	41.5034	0.9982
Trees	8×8	0.06-0.09	1,00,065	34.983	0.9883
	16×16	0.01-0.01	84,124	37.1198	0.9925
	32×32	0.01-0.1	69,110	38.5062	0.9948
	64×64	0.01-0.1	55,622	39.4841	0.9960
	128×128	0.01-0.1	51,164	40.0025	0.9965

Table 4.2: Original image and stego images which are embedded to their maximum capacities along with the threshold used and the obtained performance parameters.

Image	BS_{min}	Threshold	Capacity	PSNR	SSIM
Baboon	8×8	0.01-0.04	3,06,251	30.7401	0.9672
	16×16	0.01-0.1	2,56,505	32.3282	0.9763
	32×32	0.01-0.1	2,02,065	33.9103	0.9837
	64×64	0.01-0.1	1,31,154	36.2376	0.9916
	128×128	0.01-0.1	89,811	38.6675	0.9955
	256×256	0.01-0.1	57,854	40.457	0.9966
Building	8×8	0.05	5,18,172	42.6411	0.9957
	16×16	0.01-0.03	4,94,544	45.5285	0.9979
	32×32	0.01-0.04	4,64,115	47.0716	0.9986
	64×64	0.01-0.07	4,22,792	48.3827	0.9989
	128×128	0.01-0.1	3,88,255	49.1563	0.9991
	256×256	0.01-0.1	3,39,459	50.3012	0.9994
F16jet	8×8	0.07	4,92,402	38.6627	0.9708
	16×16	0.07	4,55,427	40.5412	0.9759
	32×32	0.01-0.09	4,02,548	41.6010	0.9808
	64×64	0.01-0.1	3,45,363	42.5485	0.9856
	128×128	0.01-0.1	2,86,582	43.5300	0.9896
	256×256	0.01-0.1	2,37,512	44.2989	0.9927
House	8×8	0.07	4,54,163	37.0667	0.9859
	16×16	0.07-0.08	4,54,163	37.0667	0.9859
	32×32	0.01-0.1	3,66,893	40.8292	0.9926
	64×64	0.01-0.1	3,20,480	41.5977	0.9938
	128×128	0.01-0.1	2,63,928	42.3119	0.9955
	256×256	0.01-0.1	2,37,312	42.7519	0.9962
Lena	8×8	0.04	4,73,291	36.6113	0.9940
	16×16	0.01-0.04	4,31,378	37.9620	0.9955
	32×32	0.01-0.06	3,76,971	38.8512	0.9964
	64×64	0.01-0.1	3,23,802	39.6424	0.9969
	128×128	0.01-0.1	2,69,667	40.4414	0.9974
	256×256	0.01-0.1	2,41,407	40.8348	0.9976
Trees	8×8	0.1	4,25,298	33.5262	0.9778
	16×16	0.01-0.09	3,73,964	34.583	0.9819
	32×32	0.01-0.1	3,02,989	35.5802	0.9854
	64×64	0.01-0.1	1,88,010	37.6281	0.9912
	128×128	0.01-0.1	73,435	41.5517	0.9966
	256×256	0.01-0.1	32,075	46.0155	0.9987

Table 4.3: Original image and stego images which are embedded to their maximum capacities along with the threshold used and the obtained performance parameters.

Original image	256 × 256 image					512 × 512 image				
	Threshold	Capacity	Stego image	PSNR	SSIM	Threshold	Capacity	Stego image	PSNR	SSIM
	0.01	78,558		32.0056	0.9706	0.01	3,06,251		30.741	0.9672
	0.07	1,15,396		37.1237	0.9765	0.07	4,92,402		38.6627	0.9708
	0.09	1,25,076		37.7581	0.99	0.05	5,18,172		42.6411	0.9957

	0.09	1,12,449		36.4788	0.9947	0.04	4,73,291		36.6113	0.9940
	0.06	1,06,215		35.7482	0.9870	0.07	4,54,163		37.0667	0.9859
	0.07	1,00,065		34.9830	0.9993	0.1	4,25,298		33.5262	0.9778

4.4 Comparison with other approaches

The results of the presented block based transform domain steganography approach is compared with spatial as well as transform domain steganography approaches using PSNR, SSIM, and capacity. Firstly, comparison is done with several spatial domain approaches in Section 4.4.1 followed by an evaluation with transform domain steganography approaches in Section 4.4.2.

4.4.1 Comparison with spatial domain approaches

Six spatial domain approaches picked for comparison are classic LSB, SCC [24], PIT [3], ST-FMM [26], Karim's method [25] and CISSKA-LSB [29]. Stego color cycle (SCC) method embeds data in least significant bit (LSB) of all the three color channels following a predetermined cyclic order [24]. In comparison, pixel indicator technique (PIT) uses an indicator channel and two data channels [3]. Indicator channel contains information about the data bits embedded in data channels. Similarly, CISSKA-LSB hides data in either Blue or Green channel according to Red channel's LSB and encrypted secret key [29]. Another approach, steganography five modulus method (ST-FMM) divides the cover image into fixed sized blocks before embedding and disperses the secret message among these blocks for better security, while Karim's method uses a secret key to design a system with enhanced security [26, 25].

Table 4.4 and Table 4.5, respectively show the PSNR and SSIM values obtained after embedding 8 KB of data in 256×256 images using various approaches. Results for the presented steganography approach are generated using a threshold of 0.08 and minimum block size (BS_{min}) of $8 \times 8, 16 \times 16, 32 \times 32, 64 \times 64$, and 128×128. Compared to the approachs CLSB, SCC, PIT, ST-FMM, and Karim's method, PSNR values obtained with the presented steganography approach are the best for all the images except for *House*. The results with minimum block size (BS_{min}) of 32×32, 64×64, and 128×128 for *House* image are not as good as reported by others. The performance of CISSKA and the presented steganography approach are more or less the same. For three out of five images, viz. *Baboon*, *F16jet* and *House*, one is better than the other at any moment of time. However, for *Building* the presented steganography approach is better and for *Trees* CISSKA has shown better performance always. The results for *Trees* can be improved by reducing the minimum block size (BS_{min}) to 4×4 at same threshold value, i.e. 0.08. This setting enhances PSNR drastically and reports a value of 54.5008 with SSIM as 0.9998. Looking at SSIM, then clearly the presented approach is

Table 4.4: Comparative results based on PSNR with spatial domain steganography approaches. Parameter values used in the presented steganographic approach are threshold = 0.08, image size = 256×256, maximum block size (BS_{max}) = 128×128. BS_{min} = Minimum block size.

Image	CLSB [29]	SCC [24]	PIT [3]	ST-FMM [26]	Karim's method [25]	CISSKA [29]	Presented approach with BS_{min} as				
							8×8	16×16	32×32	64×64	128×128
Baboon	41.3208	33.9320	33.8367	34.4472	33.9322	42.3607	41.6751	42.7341	42.6202	41.8814	42.1763
Building	28.8451	28.8451	28.8213	40.2552	28.8451	43.4071	56.0904	56.0904	55.4416	55.6684	51.5674
F16jet	47.4882	47.4852	45.6879	40.2347	47.4902	53.1665	56.1519	53.8721	49.1923	50.6310	50.3318
House	51.1659	51.1776	47.6956	40.2518	51.1564	52.7303	55.3399	55.3399	45.5882	44.1830	44.4916
Trees	39.0436	38.5418	38.2702	39.5397	38.5421	49.7496	47.4229	46.9586	47.7410	47.1144	47.4169

Table 4.5: Comparative results based on SSIM with spatial domain steganography approaches. Parameter values used in the presented steganographic approach are threshold = 0.08, image size = 256×256, maximum block size (BS_{max}) = 128×128. BS_{min} = Minimum block size.

Image	CLSB [29]	SCC [24]	PIT [3]	ST-FMM [26]	Karim's method [25]	CISSKA [29]	Presented approach with BS_{min} as				
							8×8	16×16	32×32	64×64	128×128
Baboon	0.9953	0.9938	0.9928	0.9888	0.9937	0.9937	0.9976	0.9982	0.9979	0.9972	0.9975
Building	0.9963	0.9973	0.9948	0.9765	0.9972	0.9973	0.9999	0.9999	0.9999	0.9999	0.9997
F16jet	0.9976	0.9985	0.9964	0.9797	0.9985	0.9985	0.9982	0.9984	0.9988	0.9988	0.9988
House	0.9983	0.9990	0.9974	0.9860	0.9989	0.9989	0.9989	0.9989	0.9989	0.9988	0.9988
Trees	0.9964	0.9970	0.9956	0.9858	0.9970	0.9970	0.9994	0.9993	0.9994	0.9994	0.9994

better in most of the cases. In fact, for *Baboon* and *Building*, the obtained SSIM values are the best for all the minimum block sizes. An observation made for PSNR values using the results presented in Table 4.4 is that, with an increase in the minimum quad tree block size, PSNR value reduces in most of the cases. But for *Baboon*, this observation does not hold true.

4.4.2 Comparison with transform domain approaches

The presented steganographic approach is using the notion of variable block size in transform domain and in the absence of any work that uses this concept for textual data, a comparison is made with a approach that works using a fixed block concept in the same domain [8]. Two 512×512 images, *Lena* and *Baboon*, are chosen to compare these approaches on the basis of maximum capacity and PSNR values. Table 4.6 lists the maximum capacity in each combination and the superiority of the presented steganographic approach is clearly visible. The minimum capacity value obtained for *Lena* is 1,91,384 ($BS_{min} = 64 \times 64$) and for *Baboon* its 57,854 ($BS_{min} = 256 \times 256$). Both the capacities are better as compared to fixed block approach which obtained maximum capacity of 39,936 characters with both the images [8]. Also blocks obtained with *Lena* are highly correlated as compared to those obtained with *Baboon* for large values of BS_{min}. Hence, *Lena* image reports better maximum capacity value with higher BS_{min}. On the other hand, the scenario reverses as BS_{min} decreases because smaller blocks are more correlated in *Baboon* in comparison to *Lena*. Consequently, maximum capacity of 2,94,026 is obtained when BS_{min} is 128×128 using *Lena* as a cover image. But for *Baboon* the maximum capacity reported is 4,45,222 using BS_{min} as 8×8. Similar observation is repeated with the PSNR values shown in Table 4.7 for different secret message sizes. The obtained PSNR values are almost double in most of the cases as compared to [8]. The improvement in results reported by the presented quad tree based steganograhic approach is really great.

Table 4.6: Comparative results based on maximum capacity with fixed block transform domain steganographic approach. Parameter values used in the presented steganographic approach are threshold = 0.08, image size = 512×512, maximum block size (BS_{max}) = 256×256. BS_{min} = Minimum block size.

Image	Fixed block based method [8]			Presented approach with BS_{min} as					
	DC band	Low frequency	Middle frequency	8×8	16×16	32×32	64×64	128×128	256×256
Lena	39,936	5,120	13,312	2,56,627	2,50,639	2,23,306	1,91,384	2,94,026	2,41,407
Baboon	39,936	5,120	13,312	4,45,222	4,35,353	3,93,074	3,42,743	1,61,592	57,854

Table 4.7: Comparative results based on PSNR with fixed block transform domain steganographic approach. Parameter values used in the presented steganographic approach are threshold = 0.08, image size = 512×512, maximum block size (BS_{max}) = 256×256. BS_{min} = Minimum block size.

Image	Message size (in characters)	Fixed block based method [8]				Presented approach with BS_{min} as					
		1-LSB	2-LSB	Low frequency	Middle frequency	8×8	16×16	32×32	64×64	128×128	256×256
Lena	50	36.916	36.913	33.606	36.821	82.2805	82.2805	82.2805	82.5317	81.7287	76.3867
	100	36.914	36.906	33.604	36.821	80.5727	80.5727	80.5727	80.5852	79.9132	75.1912
	500	36.882	36.852	33.586	36.035	67.2267	67.2267	67.2267	67.0797	66.4864	64.8969
	1000	36.882	36.786	33.562	35.302	66.8351	66.8351	66.8351	66.7803	65.7471	63.5019
Baboon	50	38.234	38.227	38.235	38.019	70.7364	67.9200	63.6480	65.0503	64.6909	67.0976
	100	38.229	38.218	38.232	37.875	63.4688	65.7285	61.0999	63.0509	62.6230	63.7814
	500	38.196	38.149	38.211	36.928	55.7208	54.5541	54.1226	56.2506	56.2227	57.0804
	1000	38.195	38.063	38.183	36.028	53.7595	51.8660	51.8748	53.9919	53.310	54.4671

Chapter 5

Conclusions and Future Scope

5.1 Conclusions

The presented block based transform domain steganography approach effectively combines concepts of DCT and quad trees. The number of quad tree blocks can easily be controlled using a threshold. It is observed that minimum block size (BS_{min}) plays an important role in deciding capacity; the amount of correlation in a quad tree block is significant too. Presence of such tunable parameters helps in achieving better results as compared to any fixed block method.

The presented approach adaptively embeds the secret text depending on the chosen values of threshold and BS_{min}, thus achieves high embedding capacity. In addition, the stego image embedded to its maximum available capacity looks similar to the original image and it's difficult to discriminate between the two visually. The PSNR and SSIM reported further establish acceptance of the presented approach over several existing spatial and frequency domain steganography approaches. However, if capacity is not an issue and quality is important, then one can play with threshold values and vary BS_{min} for a given cover image to get appropriate PSNR and SSIM.

5.2 Future scope

In the presented steganography approach, all the possible combinations of constraints, used for DCT blocks (threshold, minimum block size, and maximum block size) are manually tested. As different permutation results in different sizes of adaptive blocks of high frequency coefficients, which results in variable capacity, PSNR, and SSIM values. This process can be automated by adding a module that takes the allowed ranges for capacity, PSNR, and SSIM as input and perfunctorily outputs the best constraints combination suitable for a particular instance. This addition would speed up the presented steganography approach.

The presented approach can easily pre-compute the number of characters that can be embedded within a particular cover image. This fact can be utilized in selecting an image effectively from a pool of cover images. Knowing the size of the secret message and the capacity associated with available cover images, one can go for an optimal selection. In such a scenario, embedding process results in a stego image whose capacity is maximally utilized thus leaving the least redundancy in the cover image. Moreover, choosing the smallest cover image that can accommodate the secret text fully decreases utilization of the transmission bandwidth, cost and is time efficient too.

References

[1] H.R. Perera. History of steganography. `http://hareenlaks.blogspot.in/2011/04/history-of-steganography.html`, 2011.

[2] Z.K.A Ani, A.A. Zaidan, B.B. Zaidan, and H.O. Alanazi. Overview: Main fundamentals for steganography. *arXiv preprint arXiv:1003.4086*, 2010.

[3] AA-A Gutub. Pixel indicator technique for rgb image steganography. *Journal of Emerging Technologies in Web Intelligence*, 2:56–64, 2010.

[4] A. Cheddad, J. Condel, K. Curran, and P.M Kevitt. Digital image steganogarphy: survey and analysis of current methods. *Signal Processing*, 90:727–752, 2010.

[5] T. Rabie and I. Kamel. High-capacity steganography: a global-adaptive-region discrete cosine transform approach. *Multimedia Tools and Applications*, pages 1–21, 2016.

[6] D. Ou and W. Sun. High payload image steganography with minimum distortion based on absolute moment block truncation coding. *Multimedia Tools and Applications*, 74:9117–9139, 2015.

[7] T. Rabie and I. Kamel. On the embedding limits of discrete cosine transform. *Multimedia Tools and Applications*, 75:5939–5957, 2016.

[8] S.A. El Rahman. A comparative analysis of image steganography based on dct algorithm and steganography tool to hide nuclear reactors confidential information. *Computers and Electrical Engineering*, pages 1–20, 2016.

[9] T. Rabie and I. Kamel. Toward optimal embedding capacity for transform domain steganography: a quad-tree adaptive-region approach. *Multimedia Tools and Applications*, pages 1–24, 2016.

[10] T. Jamil. Steganography: the art of hiding information in plain sight. *IEEE potentials*, 18(1):10–12, 1999.

[11] A. Kumar and Km. Pooja. Steganography-a data hiding technique. *International Journal of Computer Applications*, 9(7):19–23, 2010.

[12] F. Garzia. *Handbook of Communications Security*. WIT Press, 2013.

[13] 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*, 13(4), 2004.

[14] M. Hussain and M. Hussain. A survey of image steganography techniques. *International Journal of Advanced Science and Technology*, 54:113–124, 2013.

[15] D. Seth, L. Ramanathan, and A. Pandey. Security enhancement: combining cryptography and steganography. *International Journal of Computer Applications (0975-8887) Volume*, pages 3–6, 2010.

[16] C.K. Chan and L.M. Cheng. Hiding data in images by simple lsb substitution. *Pattern Recogn*, 37(3):469–474, 2004.

[17] M. Juneja and P.S. Sandhu. Improved lsb based steganography techniques for color images in spatial domain. *IJ Network Security*, 16(6):452–462, 2014.

[18] K.H. Jung and K.Y. Yoo. Steganographic method based on interpolation and lsb substitution of digital images. *Multimedia Tools and Applications*, 74(6):2143–2155, 2015.

[19] X.Liao and C. Shu. Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels. *Journal of Visual Communication and Image Representation*, 28:21–27, 2015.

[20] C.M. Wang, N.I. Wu, and M.S Wang. A high quality steganographic method with pixel-value differencing and modulus function. *Journal of Systems and Software*, 81:150–158, 2008.

[21] HongW. Adaptive image data hiding in edges using patched reference table and pair-wise embedding technique. *Information Sciences*, 221:473–489, 2013.

[22] J. ChenW, C.C. Chang, and T.H.N. Le. High payload steganography mechanism using hybrid edge detector. *Expert Systems with applications*, 37:3292–3301, 2010.

[23] HongW and T.S. Chen. A novel data embedding method using adaptive pixel pair matching. *IEEE Transactions on Information Forensics and Security*, 7:176–184, 2012.

[24] K. Bailey and K. Curran. An evaluation of image based steganography methods. *Multimedia Tools and Applications*, 30:55–88, 2006.

[25] K. Muhammad, Z. Jan, J. Ahmad, and Z. Khan. An adaptive secret key-directed cryptographic scheme for secure transmission in wireless sensor networks. *Technical Journal, University of Engineering and Technology (UET) Taxila, Pakistan*, 20(3):48–53, 2015.

[26] F.A. Jassim. A novel steganography algorithm for hiding text in image using five modulus method. *International Journal of Computer*, 72, 2013.

[27] C. Qin, C-C Chang, and T-J Hsu. Reversible data hiding scheme based on exploiting modification direction with two steganographic images. *Multimedia Tools and Applications*, 74:5861–5872, 2015.

[28] K. Muhammad, M. Sajjad, I. Mehmood, S. Rao, and S.W. Baik. A novel magic lsb substitution method (m-lsb-sm) using multi-level encryption and achromatic component of an image. *Multimedia Tools and Applications*, (75):14867–14893, 2016.

[29] K. Muhammad, J. Ahmed, N.U. Rehman, Z. Jan, and M. Sajjad. Cisska-lsb: color image steganography using stego key-directed adaptive lsb substitution method. *Multimedia Tools and Applications*, 76:8597–8626, 2017.

[30] T. Tuncer and E. Avci. A reversible data hiding algorithm based on probabilistic dna-xor secret sharing scheme for color images. *Displays*, 41:1–8, 2016.

[31] F. Alturki and R. Mersereau. Secure blind image steganographic technique using discrete fourier transformation. In *International Conference on Image Processing*, volume 2, pages 542–545. IEEE, 2001.

[32] M. Saidi, H. Hermassi, R. Rhouma, and S. Belghith. A new adaptive image steganography scheme based on dct and chaotic map. *Multimedia Tools and Applications*, pages 1–18, 2016.

[33] A. Pandey, B. Singh, B.S. Saini, and N. Sood. A joint application of optimal threshold based discrete cosine transform and ascii encoding for ecg data compression with its inherent encryption. *Australasian Physical and Engineering Sciences in Medicine*, 39(4):833–855, 2016.

[34] M. Ghebleh and A. Kanso. A robust chaotic algorithm for digital image steganography. *Communications in Nonlinear Science and Numerical Simulation*, 19(6):1898–1907, 2014.

[35] P.Y Chen and H.J Lin. A dwt based approach for image steganography. *International Journal of Applied Sciences*, 4(3):275–290, 2006.

[36] W-Y Chen. Color image steganography scheme using dft, spiht codec, and modified differential phase-shift keying techniques. *Applied Mathematics and Computation*, 196:40–54, 2008.

[37] A.A. Attaby, M. F.M. M Ahmed, and A.K. Alsammak. Data hiding inside jpeg images with high resistance to steganalysis using a novel technique: Dct-m3. *Ain Shams Engineering Journal*, 2017.

[38] E. Avci, T. Tuncer, and D. Avci. A novel reversible data hiding algorithm based on probabilistic xor secret sharing in wavelet transform domain. *Arabian Journal for Science and Engineering*, 41:3153–3161, 2016.

[39] T.D. Nguyen, S. Arch-int, and N. Arch-int. An adaptive multi bit-plane image image steganography using block data-hiding. *Multimedia Tools and Applications*, 75:8319–8345, 2016.

[40] T. Morkel, J.H. Eloff, and M.S. Oliver. An overview of image steganography. In *ISSA*, pages 1–11, 2005.

[41] W-C. Cheng and M. Pedram. Chromatic encoding: a low power encoding technique for digital visual interface. *IEEE Transactions on Consumer Electronics*, 50(1), 2004.

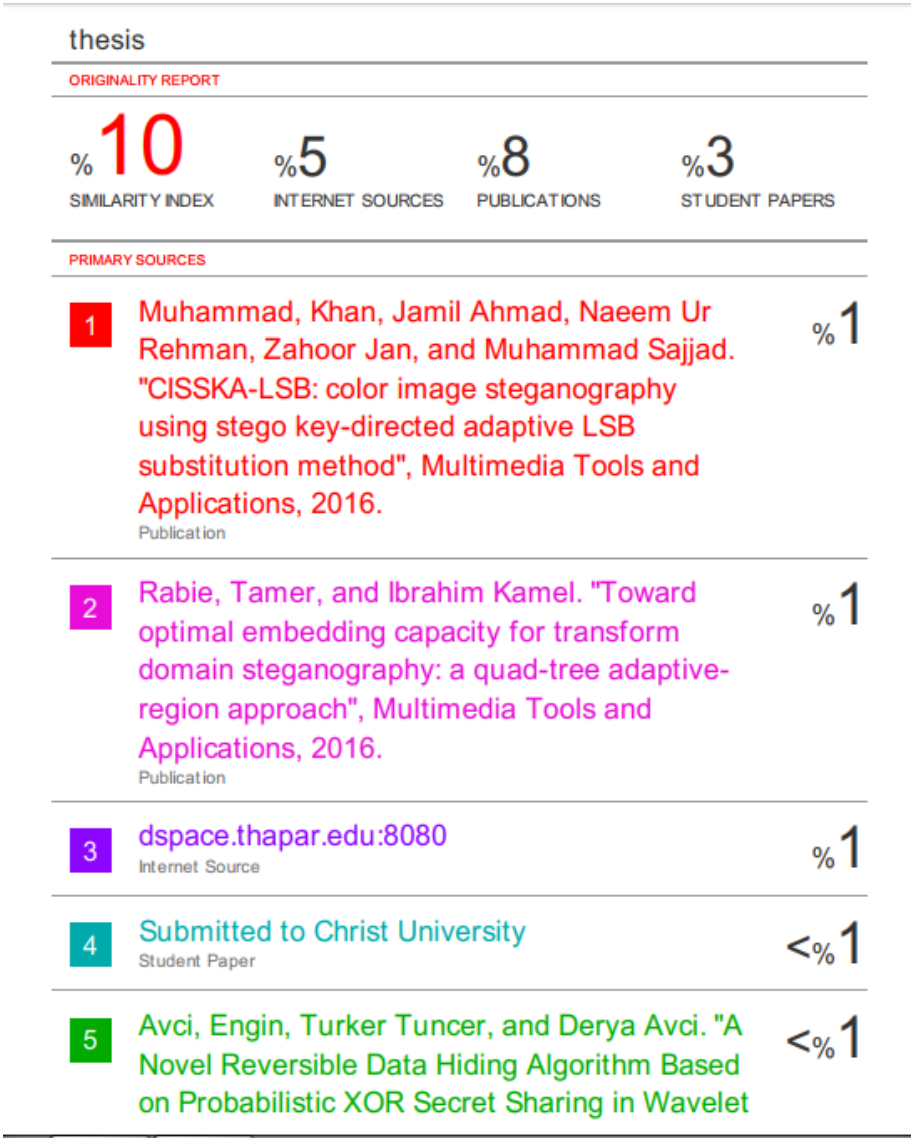
List of Publications

Jobanjeet Kaur, Shreelekha Pandey, "An adaptive quad tree based transform domain steganography for textual data", International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS 2017), Paper Code: SKR-IEEE-ECDS-1083, Tamil Nadu, India, August 2017 (Accepted).

Video link

https://www.youtube.com/channel/UCU6_rPJcJzHaxkD2EEvSdRw

Plagiarism Report



thesis

ORIGINALITY REPORT

%10	%5	%8	%3
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Muhammad, Khan, Jamil Ahmad, Naeem Ur Rehman, Zahoor Jan, and Muhammad Sajjad. "CISSKA-LSB: color image steganography using stego key-directed adaptive LSB substitution method", Multimedia Tools and Applications, 2016. Publication	%1
2	Rabie, Tamer, and Ibrahim Kamel. "Toward optimal embedding capacity for transform domain steganography: a quad-tree adaptive-region approach", Multimedia Tools and Applications, 2016. Publication	%1
3	dspace.thapar.edu:8080 Internet Source	%1
4	Submitted to Christ University Student Paper	<%1
5	Avci, Engin, Turker Tuncer, and Derya Avci. "A Novel Reversible Data Hiding Algorithm Based on Probabilistic XOR Secret Sharing in Wavelet	<%1