Predictive Reversible Data Hiding Schemes for Enhanced Embedding Capacity

A. H. M. Kamal

ID: 0411054002

A dissertation submitted in partial fulfillment of the requirement for the

degree of Doctor of Philosophy
in Computer Science and Engineering

Department of Computer Science and Engineering

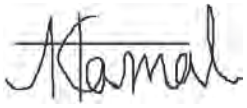
Bangladesh University of Engineering and Technology (BUET)

Bangladesh

September 2017

Declaration

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.



A. H. K. Kamal Date: 27-09-2017

Dedicated to my parents for all their love and inspiration

CERTIFATE OF APROVAL

The thesis titled "**Predictive Reversible Data Hiding Schemes for Enhanced Embedding Capacity**" submitted by A. H. M. Kamal, Student ID: 0411054002 and Session: April 2011, to the Department of Computer Science and Engineering (CSE), Bangladesh University of Engineering and Technology (BUET), has been accepted as satisfactory in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Computer Science and Engineering and approved as to its style and contents. Examination held on 27 September 2017.

BOARD OF EXAMINERS

Dr. Mohammad Mahfuzul Islam Professor, Department of CSE, BUET, Dhaka 1205	Chairman (Supervisor)
Dr. Mohammad Sohel Rahman Professor and Head, Department of CSE, BUET, Dhaka 1205	Member (Ex-Officio)
Dr. Manzur Murshed Professor, Faculty of Science and Technology, Federation University Northways Road, Churchill VIC 3842 Australia	Member
Dr. Mahmuda Naznin Professor, Department of CSE, BUET, Dhaka 1205	Member
Dr. Md. Monirul Islam Professor, Department of CSE, BUET, Dhaka 1205	Member
Dr. Mohammad Rashedur Rahman Professor, Department of ECE, North South University, Dhaka-1229	Member
Dr. A. B. M. Alim Al Islam Associate Professor, Department of CSE, BUET, Dhaka 1205	Member
Dr. Guojun Lu Professor, Faculty of Science and Technology, Federation University Northways Road, Churchill VIC 3842 Australia	Member (External)
Dr. Mohammad Nurul Huda Professor, Department of Computer Science and Engineering United International University, Dhaka-1209	Member (External)

Acknowledgements

I would like to express my sincere gratitude and profound indebtedness to my supervisor Mohammad Mahfuzul Islam for his constant guidance, insightful advice, helpful criticism, valuable suggestions, commendable support, and endless patience towards the completion of this thesis. I feel very proud to have worked with him. Without his inspiring enthusiasm and encouragement, this work could not have been completed.

I am grateful to the doctoral committee members of my thesis- Manzur Murshed of Federation University, Australia, Mohammad Rashedur Rahman of North South University, Bangladesh and Mahmuda Naznin, Mohammad Sohel Rahman, Mohammad Monirul Islam and A. B. M. Alim-Al-Islam of the Department of Computer Science and Engineering (CSE) of Bangladesh University of Engineering and Technology (BUET) for their constant support, help and assistance. I am also grateful to the other examination committee members of my thesis- Guojun Lu of Federation University, Australia and Mohammad Nurul Huda of United International University, Bangladesh.

I thank all of my teachers of the Department of Computer Science and Engineering for supporting mentally, providing advice time to time, encouraging me in all the ways to the completion of my thesis. I would also like to extend my thanks to all the staffs as well as the graduate students of the Department for their support and encouragement.

I am in debt to the Information and Communication Technology Division of the Ministry of Post, Telecommunication and Information Technology of the government of the Peoples Republic of Bangladesh for supporting me financially in form of Fellowship.

I wish to express my gratitude to Bangladesh University of Engineering and Technology for providing an excellent environment for research. The support I have received from Jatiya Kabi Kazi Nazrul Islam University in terms of approving study leave is gratefully acknowledged.

My heartfelt thanks go to my wonderful wife Afruza Sultana for her love, care, patience, and understanding during this work, and without whose encouragement and moral support this

dissertation would have been impossible. I would also like to thank my lovely son Azman Kamal, daughter Areeba Kamal, my parents, and all of my relatives and friends for their sacrifice and inspiration.

Last, but by no means least, I thank God for the talents and abilities I was given that made it possible to undertake this research.

Abstract

In reversible image steganographic schemes, an embedding algorithm implants the secret bits into a few high-frequency contents in its embedding space such that a de-embedding algorithm can extract the secret and reconstruct the original image. Most of these schemes increase the quantity of these embeddable contents through some pre-processing mechanisms like computing the transformed coefficients and measuring prediction errors by applying a prediction process. The prediction error based schemes are able to implant a higher number of bits, because the prediction errors are mostly distributed in zero or around the zero in the prediction error histogram. The embedding performance of these schemes depends on their ability to maximize the quantity of these high-frequency embeddable errors. Towards improving the frequency of these embeddable errors, this thesis aims in its first stage to improve the prediction accuracy of the existing multi-block centre reference based predictor by more rationally weighing the pixels in the prediction rules. The thesis also improves the frequency of embeddable errors by applying multiple predictors and computing the optimal error for each pixel from these multiple prediction errors or from a set of hybrid errors generated by applying these errors in a set of linear equations. The thesis demonstrates a policy of repeatedly implanting secret bits in the embeddable errors. The proposed methods of generating embeddable content for each pixel and of controlling the generation of the embeddable contents according to the demand of the application have further enriched the arena of data hiding technology. The thesis also contributes in enhancing the embedding capacity by applying prediction methodologies in the image distortion based reversible processes. It strengthens the security of the implanted data by encapsulating the implemented security levels. All the proposed schemes have the potential to make significant contributions towards hiding the large volume of data as well as to demonstrate the superior performances over their competing ones. The contribution of this thesis will hasten the arrival of new digital communication era in securing the transmission of copyrights, evidence, investigation reports, scientific results and political documents in the area of medical, forensic, law-enforcing agencies and military use.

Abbreviations

BPP	Block pixel predictor
bpp	Bits per pixel
DCT	Discrete cosine transformation
EPP	Errors per predicting pixel
FFT	Fast Fourier transform
gBL	Generalized Benford's Law
HAM	Histogram association and mapping
HDAP	Histogram of differences of adjacent pixels
LBP	Local binary pattern
LDP	Local derivative pattern
LSB	Least significance bit
LSBs	Least significance bits
LTP	Local ternary pattern
LTrP	Local tetra pattern
MBCR	Multi block center reference
ML	Multilayer
MLDC	Multilayer double cycles
MLMC	Multilayer multi-cycle
MLOC	Multilayer octuple cycles
MLPC	Multilayer pentadruple cycles
MLQC	Multilayer quadruple cycles
MLSC	Multilayer single cycle
MLSEPC	Multilayer septuple cycles
MLSEXC	Multilayer sextuple cycles
MLTC	Multilayer triple cycles
MSBs	Most significant bits
MSE	Mean-square-errors

nNEP	Number of different negative valued embeddable error points
nPEP	Number of different positive valued embeddable error points
PEBHAM	Prediction error based histogram association and mapping
PEH	Prediction error histogram
PSNR	Peak signal to noise ratio
RDE	Reversible data embedment
RDH	Reversible data hiding
RPBHAM	Repeated prediction error based histogram association and mapping
SBCR	Single block center reference
SDIM	Structural dissimilarity index
SL	Single layer
SPAM	Subtractive pixel adjacency matrix
SPP	Single pixel predictor
SSIM	Structural similarity index
SVM	Support vector machine
UIQI	Universal image quality index

Key Publications Based on PhD Research

[1] Kamal A. H. M. and Islam M. M., "Enhancing embedding capacity and stego image quality by employing multi predictors", *Journal of Information Security and Applications*, 32: 59-74, February 2017.

[2] Kamal A. H. M. and Islam M. M., "Enhancing the performance of the data embedment process through encoding errors", *Journal of Electronics*, 5(4): 79-95, November 2016.

[3] Kamal A. H. M. and Islam M. M., "Boosting up the data hiding rate multi cycle embedment process", *Journal of Visual Communication and Image Representation*, 40: 574-588, July 2016.

[4] Habiba S., Kamal A. H. M. and Islam M. M., "Enhancing the robustness of visual degradation based HAM reversible data hiding", *Journal of Computer Science*, 12(2): 88-97, March 2016.

[5] Kamal A. H. M. and Islam M. M., "Facilitating and securing offline e-medicine service through image steganography", *Healthcare Technology Letters,* 1(2): 74-79, June 2014.

[6] Kamal A. H. M. and Islam M. M., "Enhancing the embedding payload by handling the affair of association and mapping of block pixels through prediction errors histogram", in *Proceedings of International Conference on Networking, Systems and Security* (NSysS)*,* BUET, Dhaka, 5-8 January, 2016.

[7] Kamal A. H. M. and Islam M. M., "Capacity improvement of reversible data hiding scheme through better prediction and double cycle embedding process", in *Proceedings of IEEE International Conference on Advanced Networks and Telecommunications Systems* (ANTS)*,* Kolkata, India, 16-18 December 2015.

Contents

Chapter 1

Introduction

The researchers are incessantly devoting their efforts to demolish the goals of the intruders as an adversary; however, the hackers and predators are continuously seeking alternate ways to break the security by diverse mechanisms. In the recent years, a lot of divergence events including the e-mail hacking from Hillary Clinton, a candidate for the US president in 2016 election [87], stealing 50 terabytes of classified NSA data [15], hacking the secret electronic command and passwords, e.g., stealing about 81 million US dollar from the reserved of Bangladesh bank [30], have occurred. In spite of such menaces, the volume of secret information transmitted over the Internet is growing quickly [101, 102]. With the rising rate of the secret data communication over the public network, the threats to the communicating hush-hush information are also increasing hastily. Consequently, the importance of information security is escalating necessarily.

The methodologies of securing the information work by considering some security goals where the taxonomy of security goals includes confidentiality, integrity and availability [18] of data. The confidentiality refers to the matter of protecting data from being disclosed to any unauthorized device or person. The data integrity provides a Boolean information about whether the data is modified from any unauthorized access or not. The last taxonomy, i.e., the availability, is used to make sure the accessibility of data to a user on the basis of demands. The first two taxonomies are useless if the availability of data cannot be ensured. Besides, information does not come into use if it is not available for the access.

To meet these security goals, several methodologies, e.g., encryption, data fusion and data embedment, are adopted in the area of data security. A very common strategy of securing the information during the communication is to encrypt the data. The encryption process uses a secret key and applies a reversible algorithm, which is built by a one-way function, on the secret message owing to completely destroy the meaning of the secrets [18, 47]. The altered message is known as the ciphertext. The receiver decrypts the secrets from that ciphertext applying a secret key and an inverse encrypted function. The encryption process is a famous

method for securing data while transmitting the information over the Internet and storing data to a standalone device. The unauthorized deciphering is quite hard when the key and the algorithm are unknown, however, it is possible in many contexts when the portion of the key, the part of the text and/or their patterns are known to the attacker [18]. In such cases, the success rate of deciphering depends on the length of the secret key, the number of attempts that are made to break the encryption, the computing performance of the device and the guessing accuracy of the secret pattern. The average number of tries to break an encryption method usually rises as the length of the key grows. Nevertheless, still, the process of encrypting data by a long key cannot assure data security as the current computers with high computing performance are often used in cracking the data security methods. Therefore, alternative security mechanisms, like data fusion, watermarks and steganography, are getting attractions of the developers and the researchers during the implementation of the security features.

The data fusion process combines the different source generated information [10, 103]. The types of information may vary depending on the nature of the sources. The fusion node measures the correlation and the association between the data and estimates the prediction states for the integration purpose, and thus, the meaning of the original data is destroyed. Nevertheless, if the data is generated from a single source or if the data is of a single type, application of the data fusion process is not necessary. Besides, the fusion process increases the overhead information, which is required for defusing the individual data at the receiver end.

Data embedment is a popular method for providing data security. It implants data bits into a carrier and then transmits that carrier to a destination. The destination end retrieves the message from the carrier. The data embedment processes are of two types - watermarks and steganography. In both of these two embedment processes, the sender side uses a cover media as a carrier to implant the secret message into it. Before embedding the secret message into the media, the embedding scheme converts the message into bitstream [31, 32] or symbols [76]. An embedding algorithm, also known as the encoder or the data hider, implants either a single bit, a group of bits or a symbol of the message into a content or a group of contents [7, 8, 11]. This data implantation process modifies the cover media. The modified cover media are addressed as the stego media. The stego media is transmitted over the Internet to a destination. The receiver end de-embeds the data to extract the message from the stego media. The de-embedding algorithm is known as the decoder or the data extractor. The data embedment and de-embedment processes are depicted in Figure 1.1, where the vertical long line separates the

processes, i.e., operational blocks on the sender side from those on the receiving side. In the sender side, the embedding algorithm takes the secret message stream, implants it into the cover media using the secret key and the sender sends the conceived media, i.e., the stego media, to the receiver end. In the receiver end, a de-embedding algorithm extracts implanted message from the stego media using the secret key. Depending on the applied embedding and de-embedding algorithms, the scheme may reconstruct the cover media from the stego media.

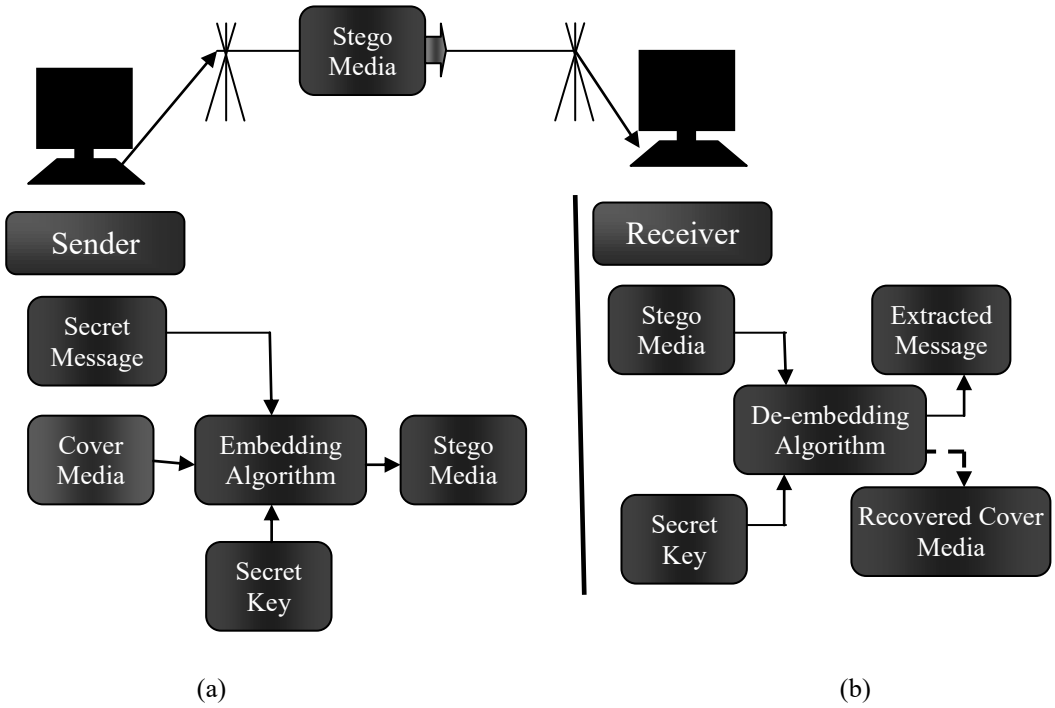


Figure 1.1: Block diagram of data hiding and data extracting processes: (a) data embedment process; and (b) data de-embedment process

Though, both the watermarks and the steganographic processes hide the message bits into a carrier, the aims of these two are different. The main purpose of the watermarking is to make sure the data integrity and identity authentication [9, 77, 92]. Hence, the watermarking policy may allow the implanted watermarks to be visible to others [100]. On the contrary, the steganographic schemes hide the message bits into the contents of the carrier in a way that the third-party cannot guess the matter of existing the secret information inside into the media. Thus, these schemes manage the confidentiality of the secrets. As a result, the steganographic schemes are becoming bewitching methods in the field of data hiding.

The distinguishing media used in the field of steganography are text [24, 65, 74, 108], audio [27, 66, 86], video [58, 109], network protocol [68, 104] and image [31, 32, 33]. Each of

the media differs from one another by their properties of having redundant information, data processing complexity and volume of data, i.e., the size of the media. The data redundancy in the media states the recurring of information in the media contents. The more the media hold redundant information, the more the media conceive message bits. The rate of redundancy also affects the distortion rate. The distortion rate decreases in the stego image when the amount of redundancy increases. Media of different formats exhibit dissimilar processing complexities in reading and writing data from/into these. That processing complexity affects the total processing speed of the data embedment process and the volume of the media impinges on the communication channels. Hence, choosing the proper media during the data embedment is an important issue.

The text holds a very limited number of redundant information and hence, cannot conceive large volume of information to meet the demand of bigger payload. Both the audio and the video contain lots of redundant information; however, these are managed by some complex algorithms and are large to be communicated over the Internet in a frequent way. The data embedment process at the header of network protocol increases network congestions. The image is the only one that, at a time, serves a set of benefits during the data embedment process. It holds much redundant information, which is the boosting property of achieving higher embedding payload and better image quality. The size of the image is also significant for embedding much information. It is enough suitable for communicating over the Internet. The images are, indeed, communicated casually and in a frequent mode over the public networks. The image as a cover media can mislead the target of the attackers to look for secret data inside it because this hides the matter of existing of the embedded data and the changes in the stego image is not detectable visually and even in many cases statistically. It is, therefore, found in the literature as the most famous carrier in the field of steganography.

Figure 1.1 highlights the process of image steganography if the word 'media' is replaced by the word 'image'. In the scheme of image steganography, the cover image is, sometimes, processed first before starting the data implantation task depending on the applied algorithm [36, 64]. The message is then embedded into the processed image. The encoder embeds a chunk of the message at each execution step. The chunk of the message consists of a bit [51, 52, 53], a group of bits [71], a numerical digit or a symbol [22, 48]. Therefore, the message is first converted into a bit stream, digits or symbols according to the demand of the applied algorithm. Each message chunk is then concealed by modifying a single content or a group of contents in an embedding space, e.g., prediction errors and discrete cosine transform (DCT)

coefficients. The embedding space is constructed from image pixels. After implanting the message, the data hider sends the stego to the destination through a public communication channel. The receiver end carries out a reverse process to extract the hidden message from the stego image.

Based on embedding space, image steganography is classified into five major groups: spatial domain, transformed domain, pixel difference domain, prediction error domain and compressed domain. In the spatial domain, message bits are directly embedded in the pixel values [106, 114]; while the transform domain undergoes a transformation of the pixel values to other formats, e.g., DCT coefficients [28], Fast Fourier transform (FFT) coefficients [82, 96], wavelet transform coefficients [45, 85], etc. An embedding algorithm implants the message chunks into these coefficients. Thereafter, an inverse transformation is done to engender the stego image. In the pixel difference domain, the embedding scheme first measures a difference between each pixel value and one of its neighbors. Figure 1.2 depicts the process where Figure 1.2(a) represents an image block. In this block, the pixels with deep gray color and white color are explored by this time, i.e., these pixels have already been processed. The pixel with black color represents the working pixel and the pixels with deep gray color are treated as immediate neighbors of the working pixel. The pixels with light gray color are yet to be processed. The value of one of the immediate neighborhood pixels is subtracted from the value of the black pixel to measure the pixel difference, as shown in Figure 1.2 (b). After computing the differences in all the pixel values, data are embedded in these different values [70, 83]. In the prediction error domain, one or several predictors are applied to predict a pixel value. As an example, the black located pixel is predicted in Figure 1.2(c) by the weighted average of its three immediate neighbors. If several predictors are applied to estimate the value of a single pixel, one of these measured values is computed as an optimal prediction value. The prediction errors are computed by deducting each predicted value from its respective cover value. These errors are used as an embedding space. The message chunks are embedded into these errors by using a set of embedding rules. After the completion of the data embedment task, these modified errors are added to the corresponding predicted values to form the stego image. Another less likely used embedding space is the compressed domain where the image data is compressed by a compression technique like vector quantization [49, 60, 95] and truncation encoded compression method [7]. These compressed data are used to implant the secret message.

The basic steganographic performance parameters are embedding payloads, media distortions due to the data implantation and the embedding time [22, 72]. The embedding payload is measured as the total implanted bits into a medium. The number of implanted bits per media content is represented by the terminology of embedding capacity, which is also known as the embedding rate. During the data implantation, the encoder modifies the contents of the media. Consequently, the cover and the stego media become statistically dissimilar. This matter of dissimilarity is termed as the media distortions. Amount of distortions is measured in several ways, e.g., by peak signal to noise ratio (PSNR), structural similarity index (SSIM). The third parameter, i.e., the time complexity, is used to measure the required times to accomplish the data embedment process. These three parameters are individually or collectively applied for comparing the performance of a scheme with the competing ones. The values of these parameters are influenced both by the types of used media and the applied embedding process.

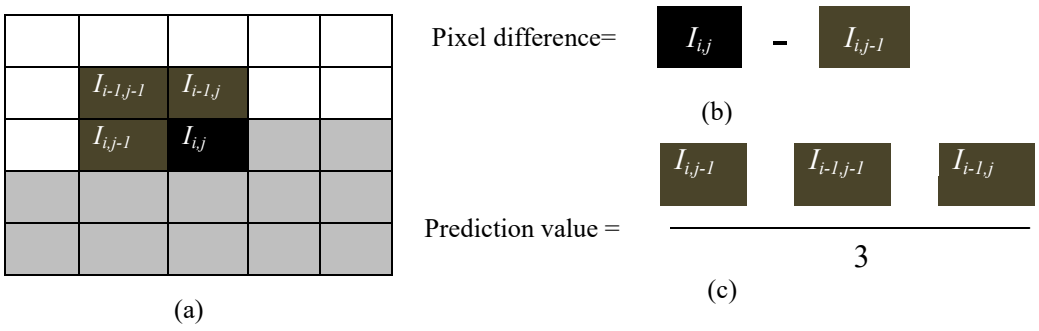


Figure 1.2: A simple method of measuring pixel differences and prediction errors: (a) an image block; (b) measurement of pixel difference; and (c) computation of predictive value.

Many schemes generate a histogram of the contents of the domain before implanting data, e.g., pixel histogram, DCT histogram, pixel difference histogram and error histogram. Several schemes compute the histogram for the image [98]; while many other schemes do the same individually for each image block [114]. During the computation of the histogram, the contents are arranged into the bins of the histogram on a scale of 0 to 255, -255 to 255 or to a user-defined range. Each bin in the histogram represents the frequency of a content that is equal to that bin value. The message chunks are embedded into the contents by shifting each bin value in the histogram [31, 114]. The histogram bins are shifted either by unequal amounts [51] or by a fixed amount for the histogram of blocky pixels [42, 43, 71].

The steganographic methodologies are classified into irreversible and reversible processes

according to their ability in reconstructing the cover image from the stego one at their data extraction phase. The irreversible techniques only focus on extracting the implanted secrets from the stego image and do not care about the reconstruction of the cover image [8, 33, 34, 48]; whereas the reversible schemes retrieve both the embedded data and the cover image from the stego one [31, 54, 58]. The irreversible schemes are inapplicable when both the extracted information and the cover contents are equally important at the data extractor end for further processing purposes. To manage the reversibility, the reversible schemes do not implant the message bits into all the contents of the embedding space; rather, these schemes conceal message bits into several highest frequency contents depending on the applied algorithm and demanded payloads [54, 113]. Many reversible schemes, additionally, implant supplementary information, e.g., the starting point of the data embedment, message length, size of image block and decoding related other secret keys/values, into a separate part of the cover image [21, 57, 94]. This supplementary information is known as the additional information/extra information/assistant information/side information/overhead information. The decoder first extracts this side information from the stego image for finding the decoding key information. The decoder, then, applies the side information to extract the secret message from the stego image and to reconstruct the cover image as well. The requirement of implanting these additional bits decreases the original message embedding capacity, known as the pure embedding capacity, and increases some degrees of processing complexity. Nevertheless, the side information increase both the security of the message and the robustness of the reversible schemes as one cannot retrieve and comprehend the secret information without realizing the meaning of these additional bits. Moreover, in many applications including medical [44, 56, 79, 89] and forensic [3, 46] applications, preserving the evidences and copyrights [43], authenticating document [5, 25] and hologram [12], ensuring security and privacy in data communication [38], performing online payment [78], electronic voting [50, 80] and trust management [105], there is no alternative for using the reversible schemes because during such critical scientific analyses the matter of retrieving both the hidden message and the original cover media are equally important. The uses of reversible applications are, therefore, necessarily increasing. When the dashed line on the right side of Figure 1.1 is considered as an active connection to a must performing process, the figure represents an abstract of the reversible scheme.

In the reversible data-hiding (RDH) arena, message bits are embedded either by shifting the histogram of the contents in the embedding space or by expanding each content

individually. The histogram manipulation based embedding processes are more famous compared with the schemes implanting bits directly into the contents of the embedding space because the content histogram provides statistics as well as a pictorial view of the content information. The statistics, inferred from the histogram, help the data hider to choose a cover image with rich embedding space, its embeddable contents and values of parameters that are used in the pre-processing stage. A very common strategy, that is used in the histogram manipulation based data embedment arena, is to implant bits into several most appeared contents. That implantation process modifies the values of the contents and thus, shifts them to other bins from their original bins in the histogram. To manage the movement of the contents during the data embedment process, these schemes select only contiguous valued contents as the embeddable contents. For this, the schemes first choose the contents of the peak presented bin from the histogram for implanting data bits. These schemes then select a sufficient number of embeddable contents from the immediate right and left positioned bins, regarding the peak one, in the histogram depending on the volume of data to be embedded. Therefore, sharper Laplacian-like distributions of the contents in the content histogram are obviously desired to meet a higher embedding capacity. The frequency of contents in n number of bins, $n>0$, varies in the histogram of the spatial values, transformed coefficients, pixel differences and prediction errors. To compare the scenario, histograms of these embedding spaces are drawn in Figure 1.3 by experimenting the well-known Lena image. The histogram, shown in Figure 1.3 (a), indicates that the highest frequency pixels of the image are at apart places. The highest appeared one contains only 500 pixels. The other three figures, i.e., Figure 1.3 (b) -1.3 (d), represent Laplacian-like distributions of their contents. The highest frequency DCT coefficient, pixel difference, prediction error shown in Figure 1.3 (b), Figure 1.3 (c) and Figure 1.3 (d) are 3200, 4000 and 8400, respectively. Thus, this is investigated in all the experimented images that the prediction error space provides the sharpest histogram. Therefore, the schemes, which implant bits into the prediction errors, present both higher embedding capacity and better stego image quality. Moreover, these schemes give stronger security of the embedded data because of their undisclosed parameters like applied predictor, the starting point of predictions in the image, the number of the associated pixels used in the prediction rules and the other parameters of the predictor own.

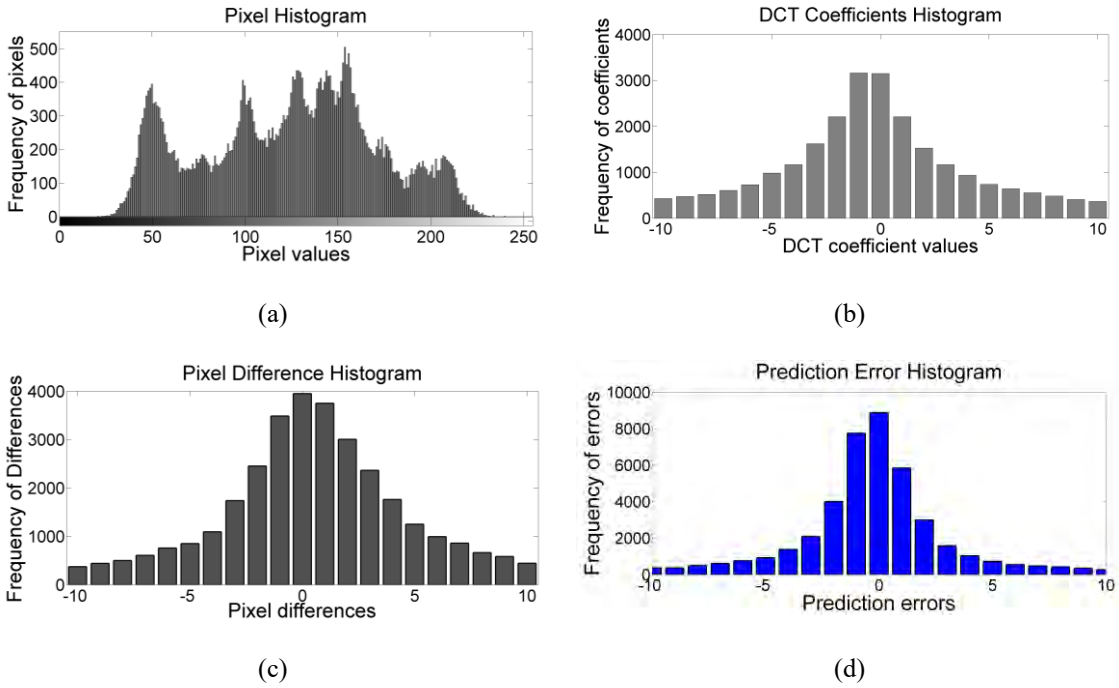


Figure 1.3: Comparison of histograms for different embedding spaces: (a) pixel value histogram; (b) DCT coefficient histogram; (c) pixel difference histogram; and (d) prediction error histogram.

In the prediction error based reversible data embedment processes, a predictor is employed to predict either a single pixel [31] or a block of pixels [32] by exploiting the features of spatial association among the neighboring pixels of the cover image. Predictors are, therefore, classified as the single pixel predictor and the block pixel predictor. Prediction accuracy depends on the correlations between the associated pixels and the predicting pixel. The size of the block also significantly affects the prediction values because when the associated pixel values in the prediction rules come from multiple blocks, the smaller sized blocks allow the predictor to predict more accurately. These associated pixels have remained unchanged during the data hiding process such that the receiving end can unambiguously predict the same values. The prediction process generates the prediction errors by subtracting the predicted values from their corresponding cover pixel values. In the recent literature, uses of multiple predictors are observed for the improvement of the prediction accuracy [63]. In that case, all the predictors separately predict each pixel value. After analyzing all the predicted values, an optimal one is chosen from these predicted values for each predicting pixel to improve the prediction accuracy. Likewise the single predictor, the prediction errors are measured by subtracting these optimal values from the corresponding pixel values.

The embedding processes are broadly classified into the single layer and multilayer data

embedment processes depending on whether they embed data bits into two [32] or more [31, 113] highest frequency errors. These high-frequency errors, to which data bits are implanted, are termed as the embeddable errors. An amount of the changes in the value of each embeddable error depends on the implanted message bit, e.g., modification amount is one unit for the implanted bit '1' and zero unit for the implanted bit '0' [32, 51]. The other errors, known as the non-embeddable errors, are certainly shifted by one unit to make sure that the non-embeddable errors will not contain the same value of modified embeddable errors. This analysis infers that the embedding capacity increases and the total image distortion decrease if the number of embeddable errors increases. The multi-layer embedding schemes [31,113], on the other hand, implant data bits into more than two highest appeared errors. These schemes embed bits into the errors of layer L in the error histogram where the embeddable errors are $\{-L, ..., -1, 0, 1, ..., L\}$, i.e., $2L+1$ different valued errors. The uses of $2L+1$ errors increase the embedding capacity but destroy the stego image quality notably. The importance of the multi-layer embedding schemes is notable when the requirement of the embedding capacity is higher, which is not achievable by using a single layer scheme. Likewise, the single layer data embedment process, both the embedding capacity and the stego image quality are improved, if the number of the embeddable error increases in the multi-layer schemes. Consequently, most schemes try to improve the embedding performance by enhancing the prediction accuracy [11, 31, 51, 63, 88]. The endeavor of improving the prediction accuracy and the attempts to enhance the frequencies of the embeddable errors are not the same. None of the schemes yet focuses directly to enhance the frequencies of the embeddable errors. Again, in the prediction error space, embedding capacity that is achievable by employing errors of $L+1$ layers is nominally higher than the one of the errors of layers L when the value of L is large because most of the prediction errors are distributed in and around the 0 valued error in the prediction error histogram. As a result, embedding into a smaller set of errors for several times, e.g., into the errors of layers L/t for t times, will increase the embedding capacity as well as the image quality rather than embedding into the errors of layer L for a single time.

Though most of the reversible data embedding schemes try to lower the image distortions [13, 55], in the recent literature, several schemes are found where distortion of cover image quality is made intentionally [52, 110]. Both the approaches have different applications in the field of the data embedment arena. In very usual cases, the embedding schemes try to reduce the image distortions in the stego image owing to keep the originality of the cover image used. The objective of such attempt is to avoid the attraction of the intruders to the transmitted stego

image by influencing them to believe that the image is not modified in any way. In many applications in the field of forensic, medical, military, satellite and industry control units, the carrier of the message is treated as another secret of the system. For the secrecy of the cover image, these applications damage the cover information in the stego image so that any third party cannot realize the cover values from the transmitted image. At the receiver end, the original cover image is reconstructed by a reversible mechanism. The destruction of the cover image is performed by two ways - (i) the quality of the image is destroyed first by a policy (e.g., encrypting the image) and the data are embedded into that destroyed contents [110]; and (ii) the embedding rules destroy the image quality during the data implantation process [71]. As all the cover information are destroyed in the stego image, the embedding process is allowed to change any number of bits of each image pixel to their binary values. If the scheme allows for changing all the bits of pixels, the embedding capacity is achievable up to 8 bits per pixel (bpp). Nevertheless, none of these existing schemes are able to give an embedding capacity of 8bpp due to the necessity of managing the reversibility.

The objective of this thesis is to contribute to the processes of enhancing the embedding capacity and the stego image quality. Focusing these objectives, the thesis, thus, emphasizes the following major aims:

i) To improve the prediction accuracy, so that the frequency of the embeddable errors is increased.

ii) To apply multi-times, say t times, data embedment into the errors of layer L/t, for $L>0$, for satisfying larger embedding capacity than embedding into the errors of layer L for a single time in the traditional multi-layer schemes.

iii) To generate embeddable codes for all the pixels or to produce only a sufficient number of embeddable codes through applying two pattern matching operators, namely, the local binary pattern (LBP) and the local ternary pattern (LTP).

iv) To enhance the embedding capacity of the distortion based embedding schemes by implanting bits into each bit of the binaries of the pixel values.

v) To improve the data security by encrypting the image before and after the data embedment as well as implanting a variable number of bits in the pixel values.

vi) To increase the embedding capacity of histogram association and mapping (HAM) based schemes by implementing the HAM policy using prediction error histogram rather than the traditional pixel value histogram.

To achieve the aforementioned aims, this thesis has reviewed a good number of latest articles on the reversible data-hiding arena. The thesis has criticized several reviewed articles and proposed their solutions as well as introduced several new methodologies for improving the embedding capacity and the stego image quality. The author of this thesis has first developed mathematical models for all the proposals. These models are then implemented in the MATLAB for the justification of novelty. The major novel contributions of this thesis are listed in the following.

i) The limitation of several schemes for the requirement of partitioning the cover image into 3x3 pixel blocks [32] has been solved through generalizing the block size to $d \times d$, for $d \geq 3$, and proposing the methodologies of predicting pixels for such block size.

ii) The quantity of the embeddable errors is increased through proposing several techniques in this thesis: (a) the prediction accuracy of the reference pixel based predictor [32] is improved by weighting the associated pixels in the prediction rules with their Euclidean distance value from the predicting one; (b) the biases affairs of the predicting pixels toward their block centre, i.e., central tendency of the block pixels, during the prediction of a pixel is analyzed in this thesis and a block centre biasing tendency parameter is incorporated in the prediction rules to further improve the prediction accuracy; (c) multiple predictors are applied in predicting each pixel to improve the number of the embeddable errors of -1 and 0; (d) incorporation of LBP codes to generate embeddable codes for all the image pixels is for the first time; and (e) an LTP based method is proposed to generate just the required number of embeddable codes by analyzing all the image pixels and the embedding rules are designed to improve the image quality by retaining the non-embeddable codes as unchanged.

iii) It is explained that the frequency of a smaller valued prediction error is higher than the larger valued prediction error. Based on this concept, the embedding capacity and the image quality are enhanced through deploying embedding rules into a fewer number of the highest frequency errors for multiple times rather than implanting into a large set of errors for a single time.

iv) Findings in the reviewed HAM based schemes state that the amount of image distortion and the number of embedded bits depend on the range value of the block pixels. Prediction error histogram based two new HAM schemes are proposed and

implemented in this thesis to meet higher embedding capacity.

The remaining chapters of this thesis are organized as follows:

In Chapter 2, a brief review of the different techniques used in predicting single pixel by applying both a single predictor and multiple predictors, the means of estimating block pixels, the common methods of implanting bits in a single layer and multi-layer data embedment processes, the way of translating blocks in HAM based schemes as well as the procedures of embedding into encrypted image are outlined. The methods of measuring the image quality and testing the security of implanted data are also epitomized.

Chapter 2 is devoted to a brief description of the techniques, which are used to improve the embedding capacity and the stego image quality in state-of-the-art technologies. As the thesis puts emphasis on improving the embedding capacity through implanting secret bits into prediction errors, a good number of prediction error based embedded schemes are concisely presented in this chapter. The predictors of different schemes are presented thereby categorizing them in some classes. The embedding rules and data extraction processes are outlined as a template for the reviewed schemes. Though most schemes try to manage better stego image quality, some of the recent schemes destroy the stego image quality intentionally for ensuring the security of cover contents. As the thesis has also contributed to this distortion based embedding area, several image distortions based schemes are also narrated in this chapter. The image quality measurement policies and the steganalysis that are commonly used in different schemes in the literature are also described in two separate sections of this chapter.

Chapter 3 explains a proposed work that improves the prediction accuracy while predicting the pixels of a block. The scheme improves the methodology of Hong and Chen's predictor [32] by applying more rational weights to the context pixels in the prediction rules. The improvement in the prediction accuracy aids the scheme in yielding better embedding capacity and stego image quality. The scheme further proposes a single layer, multi-cycle process for embedding into the same valued errors for multiple times. This multi-cycle embedment process boosts up the embedding capacity remarkably. The first part of this work has been published in *IEEE International Conference on Advanced Networks and Telecommunication Systems (ANTS) 2015* [40].

In Chapter 4, multiple predictors are employed to generate optimal prediction errors with the aim of enhancing the frequencies of two embeddable errors. The chapter discusses the methodology of applying multiple predictors for each pixel and the process of computing

optimal error from these multiple prediction errors. These optimal errors are used for data embedment. The generated optimal errors present very rich embedding space. The proposed work has been published in the *Elsevier Journal of Information Security and Applications*, 2016 [41].

Chapter 5 presents another proposed embedding scheme which extends the traditional multi-layer embedding process. Usually, multi-layer embedding schemes implant message bits into several highest frequency errors for a single time. The proposed scheme enhances the embedding capacity notably by embedding bits for multiple times into the errors of a very small layer, i.e., into a few errors. This novel work has been published in *Elsevier Journal of Visual Communication and Image Representation*, 2016 [39].

In Chapter 6, local ternary pattern (LTP) matching operator is employed to satisfy smaller embedding capacity. The proposed LTP based scheme is capable of controlling the generation of embeddable codes such that these are enough to accept the demanded message bits. The proposed embedding rules allow all the non-embeddable errors to remain unchanged. Consequently, the scheme presents a very high image quality. To satisfy large embedding capacity, the scheme applies local binary pattern (LBP) operator to generate LBP code for each pixel where each LBP code is embeddable. The generated LBP codes allow all the image pixels to accept the message bit equally. The achieved image quality with the uses of LTP and the embedding capacity for LBP are remarkably higher than all of its competing schemes. The article related to this chapter has been published in *MDPI Journal of Electronics*, 2016 [43].

Chapter 7 explains the image distortion based reversible embedding scheme. The scheme applies a predictor in image pixels to improve the embedding capacity of Ong et al.'s scheme [71] by translating pixel values of each block by exploiting the properties of error values and message chunk. The traditional histogram association and mapping scheme [71] translates block pixels of an image by utilizing the properties of the block pixel histogram. The number of implanted bits for such a traditional scheme depends on the range of the block pixel values. The proposed scheme implements the HAM scheme using prediction error histogram with the aim of achieving higher embedding capacity because this is expected that the range of absolute-valued prediction errors is smaller than the range of pixel values in a block. The findings of this chapter have been published in Proceedings of *International Conference on Networks Systems and Security* (NSysS), 2016 [42] and *Journal of Computer Science*, 2016 [29].

Chapter 8 discusses another proposed novel scheme in the area of destroying image quality before the start of data embedment. The scheme enhances the embedding capacity notably. The scheme establishes seven levels security features for the implanted data in a hierarchical manner.

Chapter 9 presents an analysis and a comparison of the proposed works of this thesis. The comparisons are performed on the values of embedding capacity and image quality.

Finally, in Chapter 10, some concluding remarks are drawn with reference to possible future directions of this research work.

Chapter 2

Literature Review

Data hiding schemes conceal the secret information into a medium. Among the data hiding policies, image steganography is a widely used one that hides the secrets into an image. The primary objective of image steganographic schemes is to increase the data embedment rate to satisfy the demand of ever-growing hiding data amount. Another objective of the image steganographic processes is to focus on increasing the security of the implanted data while managing the image quality and reducing the processing time. Researchers are continuously trying to improve the embedding capacity, i.e., increasing the number of conceivable contents by pre-processing the contents of the embedding space. Some schemes calculate pixel value histogram for embedding secret information into the most frequent pixels, while some others predict pixels and embed data into the most frequent predicted errors. The data security can be achieved by applying some security features at their pre-processing, post-processing or implantation phase. The matter of time complexity is not always a momentous issue, especially when small size of data is implanted in a single image and/or the data hider works in offline. Taking all the above-stated issues into account, this chapter provides a comprehensive review of various methods to explain the mechanism of predicting pixels on the ground of improving the embedding capacity, implanting the bits, destroying the image quality intentionally and measuring the level of image distortions.

2.1 Introduction

A common strategy of implanting data is to manipulate the histogram of spatial contents or the processed contents like prediction errors. In this strategy, the schemes conceal information into the contents of a single layer (SL) or multi-layers (ML) in the histogram. The SL data embedment schemes are employed when the embeddable data size is small. On the contrary, ML schemes implant large volume of data into an image. Both the methodologies modify the contents of the embedding space; however, the SL schemes exhibit better image quality after

the concealment. These processes do not raze the image quality deliberately; rather, their intention is to minimize the distortion rate in the stego image. Some recent schemes, on the other hand, intentionally destroy the cover image information before or during the data embedment process. These intentional distortion based schemes are employed when the cover image itself is secret.

A very common embedding space for quality preservation based SL and ML schemes is prediction errors. Prediction errors provide rich embedding space and the schemes find more embeddable elements in the error space for implanting information. Prediction accuracy in such a case plays a vital role in enhancing the frequency of embeddable errors. The number of embeddable errors increases with the advancement in the prediction accuracy. The prediction accuracy, however, varies for applying prediction techniques. Selection of an appropriate predictor, thus, is very important. The prediction accuracy again affects the distortion rate in the stego image because the level of distortions decreases for the improved quantity of embeddable errors [31, 32, 51, 72]. This distortion amount is analyzed by several methodologies, including mean-square-error (MSE), peak signal to noise ratio (PSNR), universal image quality index (UIQI), structural similarity index matrix (SSIM) [19, 99]. These terminologies of MSE, PSNR, UIQI and SSIM are defined in the following section 2.6.

The target of this thesis is to get better embedding capacity through improving the prediction accuracy either by exploring a better predictor or by employing the better prediction policy if the process does not use any predictor. An overview of the predictors used in various literature is presented in section 2.2. Section 2.3 of this chapter outlines the rules of implanting bits into the errors by the encoder and the processes of extracting message bits by the decoder. The schemes presented in this section concentrate more on enriching the error quantity of embeddable errors rather than modifying the embedment rules. Mechanisms of obtaining a distorted stego image, by employing intentional distortion-based schemes, are described in section 2.4. Reviews of non-predictive steganography schemes and the policies of measuring the level of distortions in the stego image with respect to the cover image are described in section 2.5 and section 2.6, respectively. Many security tools are used for testing the resistance of the schemes against attacks on the implanted data. These security tools are reviewed in section 2.7. Finally, a summary and some concluding remarks are presented in section 2.8.

2.2 Overview of Different Predictors

The predictors used in the steganographic schemes predict either a single pixel or a block of pixels as a group. In both the cases, the reviewed predictors are classified into several categories based on their uses of gradients, number of used predictors, the reference value(s) (e.g., block center, immediate neighbors and block median). A brief description of the different categories of predictors is presented in the sub-sections 2.2.1 through 2.2.3.

2.2.1 Gradient-Based Predictors

Gradient means changes of intensity of a pixel value in a specific direction, e.g., in the horizontal, vertical and diagonal. Say, the pixels of an image I of size $h \times w$ are read as $I_{i,j}$, for $1 \le i \le h$ and $1 \le j \le w$. The horizontal, vertical and diagonal gradients between two neighbor pixels are measured by $I_{i,j} - I_{i,j-1}$, $I_{i,j} - I_{i-1,j}$ and $I_{i,j} - I_{i-1,j-1}$ respectively. These gradient values are commonly used in weighted average prediction schemes or in edge detection based prediction schemes.

2.2.1.1 Weighted Average Predictors

A very traditional prediction policy is to predict the pixel values by averaging the grayscale values of the neighboring ones. Sachnev *et al.* in 2009 [81] predicted each pixel value by averaging four of its neighbor pixel values in the rhombus shape, i.e., by $P_{i,j} = \sum_{D} I_D$, where $D \in \{top, right, bottom, left\}$. In this prediction process, a single pixel among these four with high variant texture value can affect the predictor to estimate wrongly. This scheme also does not consider the directional effect of the changes in intensity. Ou *et al*. in 2013 [71] improved the predictor by proposing a mechanism for repeatedly updating the prediction accuracy. This predictor starts to estimate the pixel value from the average of its four neighbors in the rhombus shape, i.e., from $P^0_{i,j} = P_{i,j}$ where $P_{i,j}$ is the Sachnev's predicted value. In each of next repetitions, it computes the amount of improvement of the prediction accuracy, say Δ^t, at iteration number *t*, by taking a weighted average of four neighbor pixels' gradients, and updates the predicted values by using an equation, $P^{t+1}_{i,j} = P^t_{i,j} + \Delta^t$.

Wang *et al.* in 2014 [94] divided the image pixels into two groups 'O' and 'X' in a chessboard fashion for the prediction purpose. First, it predicts the values of 'O' pixels by averaging the values of the four neighboring pixels. These four pixels come from 'X' pixels. The scheme embeds message bits into the prediction errors. Similarly, it predicts the pixels of 'X' from the stego pixels of 'O' and embeds remaining data bits into these newly computed prediction errors.

2.2.1.2 Edge Detection Based Predictors

Edge detection based predictors first detect vertical, horizontal or diagonal edge that passes through the pixel. The predictors then compute the predicted values of each pixel by taking a weighted average of all neighbor pixels where values for weights are allotted based on detected edge. Yang *et al*. in 2013 [107] presented an edge detection based estimation technique by employing Sobel and Interpolation (SI) filtering mask into each sub-image of size $n \times n$. They improved the interpolation based prediction policy proposed by Luo *et al.* [62] and neighbors averaging scheme proposed by Sachnev *et al.* [81] by applying a weighted average of four rhombus pixels where weight distribution was governed by the detected edge. This scheme first measures both the horizontal and the vertical gradient responses by applying respectively a horizontal and a vertical SI filtering mask into a sub-image. A bigger value for the vertical gradient response means that a horizontal edge passes through the working pixel. Then the value of weights for two immediate horizontal pixels in the rhombus shape are allotted for more than the two vertical neighbors. The vice versa is true for the bigger value of the horizontal gradient response. After assigning the weights of $w_{i-1,j}$, $w_{i,j+1}$, $w_{i+1,j}$ and $w_{i,j-1}$ respectively for $I_{i-1,j}$, $I_{i,j+1}$, $I_{i+1,j}$ and $I_{i,j-1}$ from the analyses of vertical and horizontal gradient responses, the pixel $I_{i,j}$ is predicted by Eq. (2.1).

$$P_{i,j} = \frac{w_{i-1,j} I_{i-1,j} + w_{i,j+1} I_{i,j+1} + w_{i+1,j} I_{i+1,j} + w_{i,j-1} I_{i,j-1}}{w_{i-1,j} + w_{i,j+1} + w_{i+1,j} + w_{i,j-1}} \tag{2.1}$$

Due to performing the convolution between the SI mask and the sub-image of the same size, this scheme has the higher time complexity, which is augmented for the bigger sized mask. This also ignores the diagonal gradient response during its estimation task. Pixels in the $\lfloor n/2 \rfloor$ rows of each upmost and bottom of the image and $\lfloor n/2 \rfloor$ columns of each leftmost

and rightmost are not predicted for a mask size of $n \times n$, where $\lfloor . \rfloor$ stands for the mathematical floor in this thesis.

Hong W. [31] in 2012 introduced a median edge detector based estimation technique to improve the Tai *et al.*'s scheme [83] of predicting a pixel by the value of the previously processed pixel. This scheme uses immediate top, left and top-left pixels to estimate a pixel. Considering the top-left one as an edge pixel, the working pixel is predicted either from one of the other two or by subtracting the corner one from the sum of the other two. This prediction process being biased by a single corner pixel has reduced the prediction accuracy. The number of concern pixels is fixed to three immediate neighbor pixels in this scheme.

2.2.2 Multi-Predictors

Multi predictors based schemes employ multiple predictors either to improve the prediction accuracy or to embed for multiple times. Chen X. *et al.* in 2013 [11] used asymmetric predictors in their scheme, where prediction errors do not obey any symmetric distribution. The quantity of the smaller valued errors is notably unequal to the quantity of the higher valued errors. An asymmetric selection function chooses the prediction errors of the best asymmetric predictor. The authors also implemented a complementary embedding strategy using dual prediction errors for improving the embedding capacity. Two asymmetric predictors used in this scheme are chosen in such a way that if one gathers more errors on the right side of the peaked one in the prediction error histogram (PEH), the other one distributes more errors to the left side of it, e.g., the predictors min and max predict the minimum and the maximum values from the three immediate neighbors of the predicting pixel. This scheme first implants message bits into the highest frequency errors by shifting the PEH of the min predictor in a direction. The second predictor, say max, is then applied to that stego image to generate new prediction errors. At the second phase of the data embedment, this scheme implants into the most appeared errors after performing the modification of errors in the PEH in the opposite direction of the first attempt. This process works as if two separate machines apply the two predictors independently, i.e., a double-time data embedment into the same image. The asymmetric predictors are not always good predictors for obtaining better accuracy.

Another use of the multi-predictor is observed in the scheme of Ma *et al.* in 2015 [63], where multiple predictors separately predict each pixel. The optimal prediction value is generated by verifying the following situations-

i) Type I: If all the predicted values are equal, this value is regarded as an optimal predicted value.

ii) Type II: Else if all the predicted values are greater than the predicting pixel, the smallest predicted value is taken as an optimal prediction value.

iii) Type III: Else if all the predicted values are smaller than the predicting pixel, the greater value of the predictions is considered as an optimal predicted value.

iv) Type IV: when some predicted values are greater than and some are smaller than the predicting pixel, this pixel is deemed as non-embeddable and the prediction process skips that pixel without generating any error.

Our investigation reveals that Type IV pixels are about one-third of the total image pixels and this portion increases as the increase in the number of applied predictors. The empirical results presented in [63] for five predictors also justify the same scenario. Hence, doing so, this scheme skips lots of potential pixels in Type IV by marking them non-embeddable even when it is equal to one of the predicted values.

2.2.3 Reference Value Based Predictors

Reference value based predictors consider a value in the image as a predictive value for the others or computes the predicted value from a set of reference values. This process is used to predict either a single pixel or a block of pixels.

2.2.3.1 Block Center Reference Predictors

In the block center reference predictor, an image is first divided into blocks of size $n \times n$ each. The center of the block is considered as the reference value and termed as the basic pixel. While working on a block, the other four blocks, which are situated in the immediate top, right, bottom and left, are addressed as its neighbor blocks. Regarding a working block, the basic pixels of these four neighbor blocks are called satellite pixels.

Tsai *et al.* in 2009 [88] considered the basic pixel as a predicted value for the other pixels in the block. The error residues were computed by subtracting the value of the basic pixel from all the other pixels. These residues were used to conceal data bits. The basic pixel was remained unchanged and did not conceive any message bit. For bigger sized blocks, especially in texture and gradual contrast images, the accuracy of this predictor is poor.

In 2010, Hong and Chen [32] improved the prediction accuracy by associating four satellite pixels and a basic pixel of the working block in the prediction rules. A set of limitations is investigated in this prediction policy. The scheme fixes the block size to 3x3. The scheme also avoids the border pixels and all the basic pixels during the bit implantation process and hence keeps them unchanged. While predicting each corner pixel of the block, the scheme computes the average of the basic pixel and two nearest satellite pixels, i.e., puts equal emphasis on all these three pixels, although the pixel to be predicted is closer to the basic one. This irrational weighted averaging policy decreases the prediction accuracy. Lu and Huang in 2014 [59] proposed a scheme, where the basic pixels of all the blocks are also used for the data embedment. After applying the [32]'s data embedment task, they collected the unchanged basic pixels and generated another image plane upon the original one. The whole prediction and embedment processes are then applied to that upper plane to embed more data. The stego pixels in the upper plane are again distributed to their actual places in the original plane. By repeating the process of generating an upper plane from the basic pixels of the immediate lower plane, they generated a pyramidal structure. This scheme presents an improved embedding capacity. Nevertheless, this process increases the time complexity by repeatedly constructing upper levels and distributing the stego pixels of the upper level to the corresponding places in the original plane.

2.2.3.2 Neighbor Pixel Reference Predictors

Tai *et al.* in 2009 [83] used immediately processed neighbor pixel as a predicted value. The scheme used the first pixel, i.e., the pixel at left-top corner of the image, as a reference value and did not change that value. The difference of this first pixel with the second one was computed. The prediction error of the third pixel is computed by considering the second pixel as a reference value and similarly, the prediction errors for all the pixels were computed. These errors were then used to implant bits. The accuracy of this predictor is very poor as it predicts an error from only a single directional neighbor.

2.2.3.3 Block Median Reference Predictors

In the block median reference predictor based processes, the block pixels are sorted first. The value of the median pixel, of these sorted pixels, is deemed as a reference value for the other pixels of the block. If multiple pixels are of the median-valued, the first accessed one is considered as a reference median value. Leung *et al.* in 2013 [51] computed the differences of

the blocky pixels with its block median value. These differences were used for both in the SL and in the ML data embedment processes. Though this scheme increases the prediction accuracy, the performance of this predictor in the bigger sized block is low.

2.2.3.4 Combined Disjoint Predictors

Some of the prediction policies combine the results of two predictors. These predictors work in a disjointed manner. One predictor predicts one set of pixel values, while the second predictor predicts the rest pixel values in the image. Chang *et al.* [7] in 2015 presented such a scheme, where a linear block center reference predictor and a neighbor reference predictor are used in the prediction process. Before applying the prediction process, they generated a block truncated coded image if the cover image is not a block truncation coded one. In the block truncation coding, first, the pixels of each block are categorized into two groups: 'above the mean' and 'the mean and below'. All the pixels belong to the group of above the mean are quantized to a single value, say *a*. Similarly, all the pixels belong to the group of the mean and below are quantized to another single value, say *b*. A bitmap (BM) is generated to track the pixels' origin, e.g., set 1 in BM for the pixels quantized with the value of *a* and set 0 for the pixels quantized with the value of *b*. These trios (a, b, BM) are used to code a block of pixels. Data embedment is performed in this truncated image. The scheme collects one *a* from each block and arranges them in a plane. This plane is then divided into several non-overlapping blocks of a defined size. The linear predictor predicts all the pixels of a block by using a single reference pixel, e.g., the block center. The residues are computed by subtracting the predicted value from all other values in the block. Message bits are then embedded through shifting the residual errors in the histogram. A deviation *d* is computed as $d=a-b$ for all the values of *a* and *b*. This process is regarded as a neighbor reference predictor. The scheme embeds the message bits again after shifting the histogram by an amount of *d*. The *b* values are updated by subtracting the stego *d* from stego *a* to preserve the block truncation coding property. Thus, a block can conceive at most two bits. Moreover, the scheme is applicable in block truncated compressed image only.

Wang *et al.* in 2013 [97] proposed a two-disjoint-predictors based scheme where they first classified the pixels into two groups: wall pixel (i.e., border pixels) and non-wall pixels. The interpolation prediction method is used to compute the interpolation value for each wall pixel. Interpolation is performed by associating two pairs of neighboring non-wall pixels. The differences between the wall pixels and their corresponding interpolation values are measured

to find interpolation errors. A part of the secret message is embedded into these interpolation errors. Again, while processing the non-wall pixels, these non-wall pixels are first arranged in a defined traversing order. In this arrangement, a difference between a pixel and its immediate earlier traversed pixel is computed. These two pixels become always neighbor in the cover image. This way, all the differences are computed. The remaining data bits are implanted into these computed pixel differences.

2.3 Quality Preservation Based Reversible Steganographic Rules

Prediction error based reversible data embedment schemes first compute the prediction errors. A reversible algorithm is applied to implant data bits into these errors. To maintain the reversibility as well as the image quality, data bits are embedded into several highest frequency errors only. The values of these higher frequency errors are called embeddable errors and the rest are addressed as non-embeddable errors.

2.3.1 Data Embedment Policies

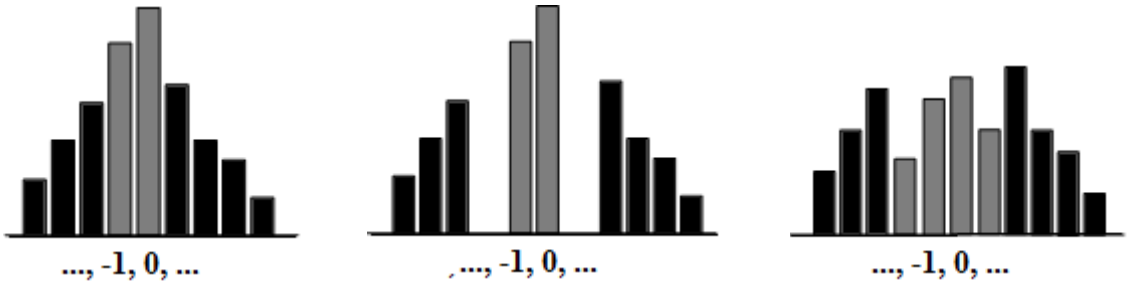
The SL data embedment processes embed message bits into the errors of {0} or {-1, 0}; while the ML schemes implant bits into the errors of L layers, for L>0, in the error histogram, i.e., into {-L, ..., -1, 0, 1, ..., L} valued 2L+1 dissimilar errors. In both the cases, the error histograms are modified or shifted to make space for implanting data, as shown in Figure 2.1 and 2.2 for the SL and the ML schemes respectively. In these processes, at first, the non-embeddable errors are shifted by an equal amount, e.g., non-embeddable positive and negative valued errors are shifted to the right and left directions respectively by an equal amount, to create sufficient vacant positions in the histogram. The embeddable errors are modified by the embedding rules. The Figure 2.1(c) and 2.2(c) represent the state of the stego errors in the SL and the ML scheme. The scenarios are almost the same for all the reviewed schemes. Consequently, the data embedment rules are defined in the following as a template. Let the predicted value of $I_{i,j}$ the pixel in the image I is $P_{i,j}$. Each prediction error $e_{i,j}$ is measured by $e_{i,j} = I_{i,j} - P_{i,j}$. The SL and ML data embedment rules implant bit into this error $e_{i,j}$ by applying the Eq. (2.2) and (2.3), respectively. Finally, the stego pixel $\tilde{I}_{i,j}$ is reformed by the Eq. (2.4).

$$\tilde{e}_{i,j} = \begin{cases} e_{i,j} - 1 & if\ e_{i,j} < -1 \\ e_{i,j} - b & if\ e_{i,j} = -1 \\ e_{i,j} + b & if\ e_{i,j} = 0 \\ e_{i,j} + 1 & if\ e_{i,j} > 0 \end{cases} \tag{2.2}$$

$$\tilde{e}_{i,j} = \begin{cases} e_{i,j} - L & if\ e_{i,j} < -L \\ 2e_{i,j} - b + 1 & if\ -L \le e_{i,j} < 0 \\ 2e_{i,j} + b & if\ 0 \le e_{i,j} \le L \\ e_{i,j} + L + 1 & if\ e_{i,j} > L \end{cases} \tag{2.3}$$

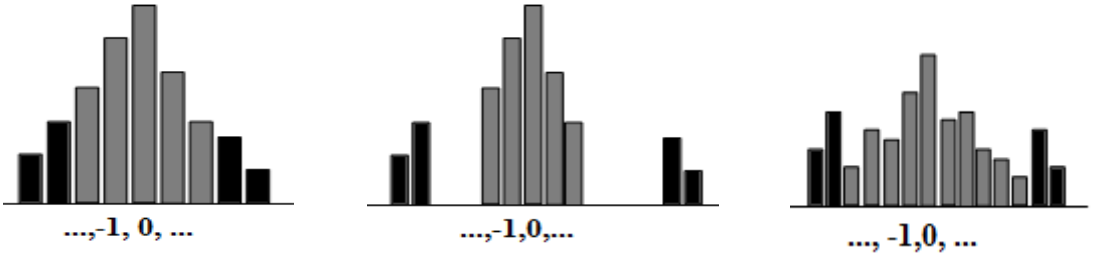
$$\tilde{I}_{i,j} = P_{i,j} + \tilde{e}_{i,j} \tag{2.4}$$

Where *b, L* and $\tilde{e}_{i,j}$ stands for a message bit, embedding layer in the histogram and stego error, respectively.



(a) (b) (c)

Figure 2.1: SL data embedment process for implanting bits into the prediction errors: (a) prediction errors; (b) shifting of non-embeddable errors; and (c) Stego errors after bit implantation. The gray color represents the embeddable errors, whereas black color to represent non-embeddable errors.



(a) (b) (c)

Figure 2.2: ML data embedment process for implanting bits into the prediction errors: (a) prediction errors; (b) shifting of non-embeddable errors; and (c) Stego error after implantation of bits. The gray color represents the embeddable errors, whereas black color to represent the non-embeddable errors.

2.3.2 Data Extraction Policies

During the data extraction, a reverse process is carried out. At first, the receiver-end computes the same predicted value, $P_{i,j}$ as computed during the embedment process by associating the same set of pixels in the prediction rules. The stego errors $\tilde{e}_{i,j}$ are computed using the Eq. (2.4). The message bit that was implanted into each error, i.e., in $e_{i,j}$ in the SL or the ML embedment process, is extracted by using the Eq. (2.5) and (2.6), respectively.

$$b = \begin{cases} 0 & if\ \tilde{e}_{i,j} = 0\ \ or\ \ \tilde{e}_{i,j} = -1 \\ 1 & if\ \tilde{e}_{i,j} = 1\ \ or\ \ \tilde{e}_{i,j} = -2 \end{cases} \tag{2.5}$$

$$b = \begin{cases} \text{mod}(\tilde{e}_{i,j}, 2) & if\ 0 \le \tilde{e}_{i,j} \le (2L+1) \\ \text{mod}(\left|\tilde{e}_{i,j}\right| - 1, 2) & if\ \text{-}1 \le \tilde{e}_{i,j} \le -2L \end{cases} \tag{2.6}$$

In the Eq. (2.6), the function mod(*a, b*) outputs the remainder value when *b* divides *a* and |.| stands for the absolute value of an expression. The function |.| carries same meaning throughout this thesis.

2.3.3 Cover Reconstruction Policies

The errors for reconstructing of cover image are computed for both the SL and the ML schemes using the Eq. (2.7) and (2.8), respectively.

$$e_{i,j} = \begin{cases} \tilde{e}_{i,j} + 1 & if\ \ \tilde{e}_{i,j} < -1 \\ \tilde{e}_{i,j} - 1 & if\ \ \tilde{e}_{i,j} > 0 \end{cases} \tag{2.7}$$

$$e_{i,j} = \begin{cases} \tilde{e}_{i,j} + L & if\ \ \tilde{e}_{i,j} < -2L \\ -\left\lfloor \left|\tilde{e}_{i,j}\right| / 2 \right\rfloor & if\ \ \text{-}2L \le \tilde{e}_{i,j} < 0 \\ \left\lfloor \tilde{e}_{i,j} / 2 \right\rfloor & if\ \ 0 \le \tilde{e}_{i,j} \le 2L+1 \\ \tilde{e}_{i,j} - L - 1 & if\ \ \tilde{e}_{i,j} > 2L+1 \end{cases} \tag{2.8}$$

Finally, the cover image is reconstructed from these retrieved errors using the Eq. (2.9).

$$I_{i,j} = P_{i,j} + e_{i,j} \tag{2.9}$$

2.3.4 Payload Volume Dependent Embedding Process

A data embedment process in some schemes [51, 94] depends on the volume of the payload, say *C*. Wang *et al.* in 2014 [94] proposed a payload volume dependent data embedment scheme, where 'rate-distortion model' is used for embedding low volume of data, while 'multi-layer model' is used to embed high-volume of data. In both the cases, at first, a histogram is drawn from the computed prediction errors on the cover image.

Rate-distortion model:

i) Find an optimal bin with a frequency f_r in the right side of the histogram so that $f_r > C$. If such bin exists, embed data bits into these bin errors. Otherwise, go to the next step.
ii) Find an optimal bin with a frequency f_l on the left side of the histogram so that $f_l > C$. If such bin exists, embed data bits into these bin errors. Otherwise, go to the next step.
iii) Checking simultaneously, find two bins with frequency f_l and f_r so that $(f_l + f_r) > C$. If such bins exist, embed data bits into these bin errors. Otherwise, go to the next step (i.e. first, step in the sub-optimal scheme).

Sub-optimal scheme: Multilayer model

i) Find two peak-presented bins with the frequencies f_l and f_r respectively on the left and right side of the error histogram. If $(f_l + f_r) < C$, measure the maximum value '*f*' of these two frequencies and the value '*e*' of the corresponding peaked error.
ii) Embed data bits into the errors of value '*e*' and calculate the remaining payload as $C = C - f$.
iii) Repeat step i and ii until $(f_l + f_r) < C$. Otherwise, go to step iv.
iv) As a final layer, apply the rate-distortion model.

2.4 Intentionally Quality Distortion Based Reversible Steganographic Process

Many schemes [52, 71, 110] destroy all the cover information in the stego image when the cover image itself is secret. This distortion is performed either during the data embedment time by the embedding rules or before starting the embedment process.

2.4.1 Razing the Cover Image While Applying the Data Embedment Rules

A simple methodology of distorting a cover image is to apply ML embedding processes into the errors of larger embedding layer L. Although, the ML processes destroy the image, notably through embedding data into the errors of larger value of L, these schemes do not destroy the cover information entirely and lots of cover information remains visible in the stego image. Ong *et al.* in 2014 [71] presented an improved cover image distorted based scheme, where the cover image is razed effectively by translating each block of pixels by an amount. The scheme first divided the cover image into k blocks. The range value R_i, for $1 \le i \le k$, of pixel values in each block i is measured. While working on each block i, that R_i is used to divide the grayscale into equal spaced P_i parts, where $P_i = 2^{8-\lceil \log_2 R_i \rceil}$ and $\lceil . \rceil$ stands for mathematical ceiling operation. The length of the embeddable message chunk, say b_i, in the block i is defined by $b_i = 8 - \lceil \log_2 R_i \rceil$. Figure 2.3(a) demonstrates an example, where the possible embeddable message chunk is one of the {00, 01, 10, 11} for $32 < R_i \le 63$, P_i=4 and b_i=2. Primarily the histogram of blocky pixels will take place at one of the P_i parts, which is known as the original partition. The second part of the left in Figure 2.3 (a) is the original partition. Each arrow line indicates the partition into which the original histogram will be reflected and each bold and italic number, i.e., the label of each arrow line, represents a possible embeddable message chunk. To carry out this reflection, at first, an association of the original partition is performed with one of the parts P_i according to message chunk b_i. Each bin of the histogram is then mapped to the corresponding position in the associated partition to perform the reflection. The methodology is called histogram association and mapping (HAM). Thus, after the embedment of message chunk '10', the original histogram, as depicted in Figure 2.3(a), is reflected in the last partition in the gray part by the HAM policy. That reflection is depicted in Figure 2.3 (b). This reflection ensures a change in the values of all the pixels in the block. The embedding capacity of the scheme depends on the range value of the block because b_i is derived from R_i and b_i increases with the decrease in the value of R_i. Hence, minimizing the value of block range R_i eventually increases the distortion of the cover image.

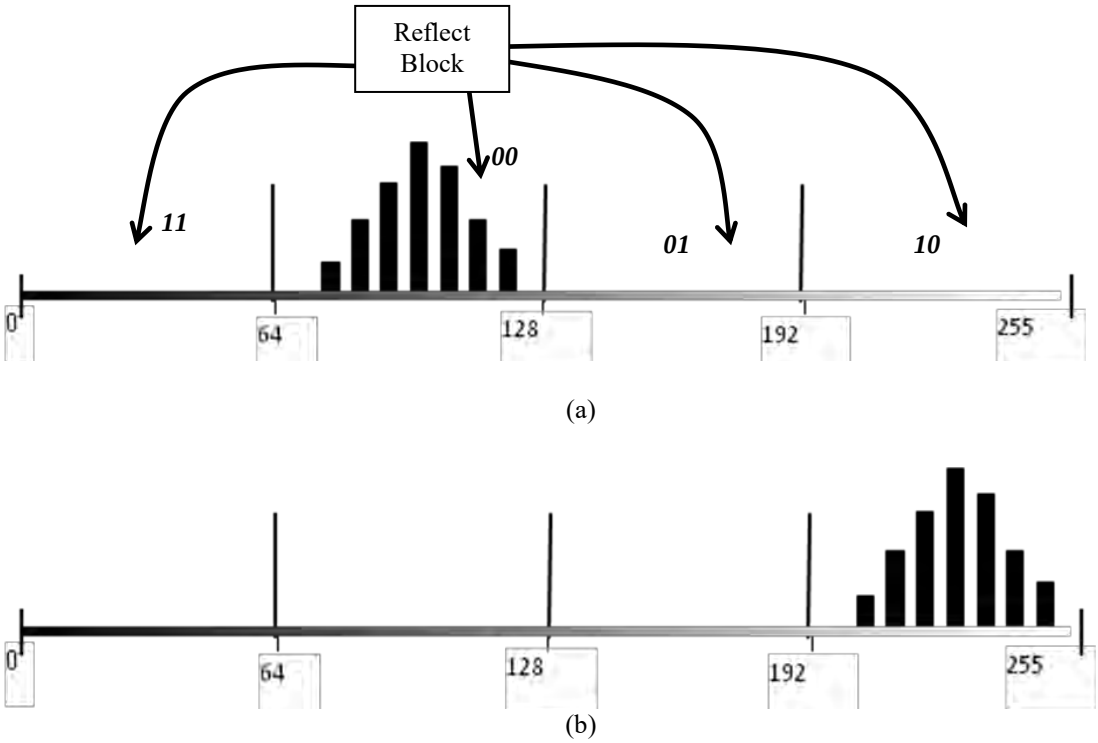


Figure 2.3: Histogram association and mapping policy: (a) position of the original and reflection partitions for $32 < R_i < 63$; and (b) Position of Stego error histogram after implanting a message chunk of '10'.

2.4.2 Razing the Cover Image Before the Data Embedment

Many schemes [52, 110] destroy the cover image before implanting data. The very famous technique of destroying the image before data implantation is to encrypt each image pixel by performing bit-wise simple *exclusive-or* (XOR) operation with a key. Liao and Shu in 2015 [52] presented an image razing data embedment scheme, where the image is destroyed, first, using the very common XOR based image encryption technique. The encrypted image is then converted into $m \times n$ sized blocks. The contents of each encrypted block are further separated into two groups S_0 and S_1 with the size of $\lfloor (m \times n)/2 \rfloor$ and $\lceil (m \times n)/2 \rceil$, respectively, where $\lfloor . \rfloor$ represents the mathematical floor operation. One bit of information is embedded into this block by flipping three least significant bits (LSBs) of all the pixels in either S_0 or S_1 depending on whether the data bit is 0 or 1. The embedding code fragment is provided in Figure 2.4.

In the extraction phase, the stego image is decrypted first by using the decryption key. Likewise the embedding phase, the decrypted image is partitioned into blocks of $m \times n$ pixels. These stego pixels are grouped into two sets $\overline{S}_0$ and $\overline{S}_1$ by size $\lfloor (m \times n)/2 \rfloor$ and $\lceil (m \times n)/2 \rceil$, respectively by the decoder. As three LSBs of pixels of S_0 or S_1 was flipped by the embedding rules, the cover block will be generated from $\overline{S}_0 \cup \overline{\overline{S}}_1$ or $\overline{\overline{S}}_0 \cup \overline{S}_1$ where $\overline{\overline{S}}_0$ and $\overline{\overline{S}}_1$ represent the three LSBs flipped pixels of the set of $\overline{S}_0$ and $\overline{S}_1$ respectively, thus, $S_0 = \overline{\overline{S}}_0$ and $S_1 = \overline{\overline{S}}_1$. Let $H_0 = \overline{S}_0 \cup S_1$ and $H_1 = S_0 \cup \overline{S}_1$. This is certain that the original cover block is one of the H_0 and H_1. The pixels in H_0 and H_1 are applied in a relation to measuring the block complexity F_0 for H_0 and F_1 for H_1. These complexities are measured by summing up the differences between each of the pixels and the mean of its neighbors. The measured complexity is used in an analysis to identify whether H_0 or H_1 belongs to the original block. If H_0 belongs to the original block, message bit '0' is extracted and the H_0 is reconstructed as the cover block. Otherwise "1" is extracted and H_1 is reconstructed as the cover block. The stated process is repeated for all the blocks. Finally, the cover image is reconstructed by concatenating all of the retrieved blocks. This scheme, however, suffers from very smaller embedding capacity as it embeds only a single bit into every block of $m \times n$ pixels.

If secret message bit is "0"

Flip 3 LSBs of all pixels in the set S_0. Say these are $\overline{S}_0$

Else if secret message bit is "1"

Flip 3 LSBs of all pixels in the set S_1. Say these are $\overline{S}_1$

End

Figure 2.4: Data implantation by flipping three LSBs of a group of pixels

In 2014, Zhang *et al.* [110] proposed another scheme which provided higher embedding capacity than the [52]. Likewise the [52], this scheme first encrypts the cover image and thereby destroys the image quality notably. All the encrypted pixels of the image are then divided into two parts, namely "Black" and "White", where black and white pixels are collected from a chessboard fashion distribution of whole encrypted pixels. The fourth LSBs of all the white pixels are extracted and then compressed. The compressed 4th LSBs along with

the secret message bits are embedded in the white pixels by replacing all of the 4th LSBs. The scheme has to embed additional information, e.g., 4th LSBs of all white pixels, and thus, reduces the actual payloads. Moreover, the black marked pixels are not used for message bit implantation.

2.5 Non-predictive Reversible Data Hiding Scheme

Pan *et al.* (in 2015) [114] proposed a non-predictive reversible data-hiding scheme. The scheme divides the cover image into blocks of size $s \times s$. Histogram of spatial values is measured individually for each block. The highest frequent pixel B_P is marked in each block histogram. If B_P is not at the extreme of the histogram i.e., not in the leftmost or the rightmost of the histogram, its immediate right and left neighboring peaks are explored at $B_P + 1$ and $B_P - 1$, respectively. If the highest peak with two neighbor peaks is still absent, the block is skipped. Otherwise, data are embedded starting from layer m = 0 following rules given below:

i) In each block, the scheme embeds message bits into pixels of $B_P + 1$ and $B_P - 1$ shifting its pixel histogram in right and left directions, respectively. It concatenates these modified blocks to form stego image at embedding layer *m*.

ii) If data bits to be embedded are still in hand, it reuses the stego image for the next embedding layer $m+1$ and partitions the stego image into blocks by doubling its dimension by $(2^m s) \times (2^m s)$, i.e. $2s \times 2s$ for m=1, $4s \times 4s$ for m=2 and so on; and followed the same procedure for embedding message bits.

The scheme does not change the peak presented pixels for the purpose of data extraction and cover image reconstruction. The layered approach works independently and thus, the process works as a multi-time data embedment method.

2.6 Methodologies for Measuring the Image Distortions

There are many ways of measuring the image distortions of a test image. The traditional measuring policies are measuring the mean square error (MSE), root means square error, mean absolute error, average difference, maximum difference, standard deviations, normalized cross-correlation, peak signal to noise ratio (PSNR) between the original and the test image [19, 93, 99]. All of these methodologies are developed based on the computation of pixel

differences. Among these policies, MSE and PSNR are widely used. Let the cover and the stego images are I and $\tilde{I}$, respectively. The MSE and the PSNR of an image with a size of $h \times w$ are measured using Eq. (2.10) and (2.11), respectively.

$$MSE = \frac{\sum_{i=1}^{h}\sum_{j=1}^{w}(\tilde{I}_{i,j} - I_{i,j})^2}{h \times w} + \varepsilon \tag{2.10}$$

$$PSNR = 10\log\frac{255^2}{MSE} \tag{2.11}$$

Where $\varepsilon = 0.65 \times 10^{-5}$ is added to the MSE to measure the 100dBm of PSNR while comparing two same images. Thus, the PSNR loss due to data implantation is measured by the Eq. (2.12).

$$PSNR\_Loss = 100 - PSNR \tag{2.12}$$

The visual attributes of an image that a human can perceive are brightness, contrast, shape and texture of objects, orientations, smoothness, etc. Since the sensitivity of the human visual system is dissimilar to a different aspect of the images; it makes sense to account for these sensitivities during the comparison of two images [93, 99]. The universal image quality index (UIQI) [93] combines the factors like loss of correlation, luminance distortion and contrast distortion. To measure UIQI, at first, the image is divided into blocks of size $d \times d$ each and then the block means and the block variances for each of the cover blocks and the stego image blocks are measured. Let μ_C and μ_S be the block means of a specific block in the cover image and the respective block in the stego image, respectively and σ_C^2, σ_S^2 be their respective block variances. Say, σ_{CS}^2 is the covariance measured for the same block of the cover and the stego image. From these three factors, UIQI of the block is measured using Eq. (2.13).

$$UIQI = \frac{4\sigma_{CS}\mu_C\mu_S}{(\mu_C^2 + \mu_S^2)(\sigma_C^2 + \sigma_S^2)} \tag{2.13}$$

The UIQI values for all the blocks are measured and a mean of these is computed to compare the image quality. The UIQI is an unstable measure when $(\mu_C^2 + \mu_S^2)$ or $(\sigma_C^2 + \sigma_S^2)$ is very close to zero. In addition, UIQI does not correlate with the subjective assessment. Nevertheless, the human visual system is highly adapted to extract structural information from visual scenes. Structural similarity index (SSIM) between two images, therefore, provides

better comparable information [16, 99]. The SSIM value for a specific block of an image is measured by using the Eq. (2.14).

$$SSIM = \frac{(2\mu_C\mu_S + c_1)(2\sigma_{CS} + c_2)}{(\mu_C^2 + \mu_S^2 + c_1)(\sigma_C^2 + \sigma_S^2 + c_2)} \tag{2.14}$$

Where, c_1 and c_2 are two constants. Values of these two constants were set to 0.01 and 0.03, respectively, for all the experiments presented in this thesis so that the divisor never becomes 0. Computing the SSIM value for all the individual blocks and then averaging them compute the SSIM value for an image.

Another visual attribute of an image, namely, structural dissimilarity index (SDIM) is a distance metric, which is measured by the Eq. (2.15).

$$SDIM = \frac{(1 - SSIM)}{2} \tag{2.15}$$

All of the attributes are the useful parameters for comparing the structural dissimilarities which are happened after implementation of the data implantation process.

2.7 Tools for Testing the Security of Implanted Data

An attempt at maintaining better image quality does not ensure the security of the implanted data because there are many steganalyzers available for detecting the hidden data. Some analyzers detect the changes in the LSBs [21,35, 75, 106], some others analyze the difference histograms to detect the modification performed [74, 110, 111]. A few of the steganalyzers analyze the dependencies among the pixels by using a Markov model in order to apply the Markovian features into a support vector machine [19, 73]. Generalized Benford's law is also used to detect stego contents by verifying the behavior of the significant digits in the images [1, 2, 23]. The standard deviation between two images and the correlation coefficients are also used as security tools to measure the modifications occurred in an image. Relative entropy is also analyzed to detect any major changes made in an image [17, 37]. Among these tools, Chi-Square test, histogram differences, generalized Benford's law, standard deviations and relative entropy were used in this thesis to test the resistance of various schemes against statistical attacks.

2.7.1 Chi-Square Test

Chi-Square test is a very illustrious process to check the pixels' modifications in the stego image with respect to the pixel values in the cover image [17, 20]. In this process, a histogram of the image pixels is computed. The frequency of every bin in the histogram is considered as an independent sample. As an image can be converted to grayscale, without loss of generality, the degree of freedom (DF) for this test was taken at 255, i.e., $DF = 255$. The frequency of each pixel in both the cover and the stego image, i.e. bin values in the histogram, is compared with the expected ones in the Chi-Square relation. The Chi-Square statistics of these pixels, whose frequencies in the histogram of stego image are greater than 4 are measured by the Eq. (2.16).

$$X^2 = \sum_{i=1}^{h}\sum_{j=1}^{w}\frac{Oberved_{i,j} - Expected_{i,j}}{Expected_{i,j}} \tag{2.16}$$

For testing the null hypothesis, a critical value is measured. The MATLAB programmer can use the tool chi2inv by setting the probability to 0.05 for measuring the critical value. Let the measured critical value is X_α^2. The test of the null hypothesis is accepted if this holds $X^2 \ll X_\alpha^2$ for all the images measured.

2.7.2 Histogram Differences

Zhao *et al*. in 2009, [111] have shown that, in the image of the archetype, there is a similarity between the changes of intensities along the vertical and horizontal direction. To check this matter, a vertical difference histogram and a horizontal difference histogram are computed by taking differences of every two neighbor pixels along the vertical and the horizontal directions, respectively. Any major dissimilarity between these two histograms is regarded as a sign of tempered image. Let the histograms of the vertical differences and the horizontal differences are $\tilde{H}_v$ and $\tilde{H}_h$, respectively. The discrepancy *D* between these two histograms for several most appeared differences, e.g., bin values in the range [-T, T], is calculated by using the Eq. (2.17).

$$D = \sqrt{\sum_{i=-T}^{T}\left(\tilde{H}_h(i) - \tilde{H}_v(i)\right)^2} \tag{2.17}$$

For all the empirical tests conducted for this thesis, the value of *T* was set to 20 to minimize the computation time. A smaller value of *D* yielded from the Eq. (2.17) typifies that the image is not tempered and, thus, this does not contain any hidden information.

2.7.3 Analyzing the Standard Deviation and Correlation Coefficient of Images

Standard deviations and correlation coefficient are measured from the cover and stego images. The divergence between the standard deviations of the cover and the stego images is 0% if the cover and the stego images coincide. The divergence in higher percentage is used to guess the existence of hidden data. For better image quality, a smaller value of divergence is expected. The value of correlation coefficient also provides statistical information on the relative similarities between the two images. When the correlation coefficient value is close to 1, two images are statistically alike. Therefore, the correlation coefficient is used as a security-testing tool.

2.7.4 Comparing the Relative Entropy

To measure the divergence of stego image (*S*) from the original image (*I*), the relative entropy provides useful information. The relative entropy (*D*) between the probability distributions of the original image (*I*) and the stego image (*S*) is calculated by using the Eq. (2.18).

$$D(I \,||\, S) = \sum_{x=0}^{255} I(x) \log \frac{I(x)}{S(x)} \tag{2.18}$$

If the relative entropy (*D*), which lies between is zero, the system is perfectly secure. *D(I||S)* is a nonnegative continuous function and equals to zero if and only if *I(x)* and *S(x)* coincide for all the *x*. Thus, *D(I||S)* can be normally considered as a distance of entropy between the measures of *I(x)* and *S(x)*.

2.7.5 Generalized Benford's Law

Steganalysis based on Benford's Law is a very latest, faster and effective mechanism to detect larger modifications in a large volume of natural data [1, 2, 23]. The Benford's law is stated from the investigation that, among the significant digits, the probability of appearing '1' as a first significant digit in a large set of natural numbers is more than the probability of appearing '2' and so on, i.e., *P(1)>P(2)>* ... ... *>P(9)*, where *P(n)* is the appearing probability of n ($1 \leq n \leq 9$) as a first significant digit. The expected values of *p(n)* in a large volume of discrete cosine transformed (DCT) coefficients can be measured by the generalized Benford's law (gBL) using the Eq. (2.19).

$$p(n) = N.\log_{10}\left(1 + \frac{1}{s + n^q}\right), n = 1, 2, ..., 9 \quad (2.19)$$

Where, N, q and s define the accuracy of the relation at different compression quality factors. A goodness-of-fit of these three parameters N, q and s are measured using Matlab curve fitting toolboxes in [1] for each of the different quality factors, say, 50, 60, 70, 75, 80, 90 and 100. The goodness-of-fit values of N, q and s are used in (2.19) to estimate the best value by it. For example, for the compression quality factor of 75, the values are fitted to N=1.396, q=1.731 and s= -0.3549.

In the experiments, the mean distributions of each significant digit i, say μ_i, in the DCT coefficients are measured. At the same time the expected distributions p_i of each digit i are computed by the Eq. (2.19). Let the mean distributions and the expected distributions in the cover and the stego images are μ_i^C, μ_i^S, p_i^C and p_i^S, respectively. The percentage of differences between the expected distributions and the mean distributions for all the digits i are measured for both the cover and the stego images as $d_i^C = \frac{\left(p_i^C - \mu_i^C\right)}{p_i^C} 100\%$ and $d_i^S = \frac{\left(p_i^S - \mu_i^S\right)}{p_i^S} 100\%$, respectively. If the changes in the stego image due to the data embedment do not happen on a large scale, the values of d_i^C and d_i^S will be very close. A difference d_i between d_i^C and d_i^S is measured by the Eq. (2.20).

$$d_i = d_i^C - d_i^S \quad (2.20)$$

The value of d_i must be smaller than a threshold T_i, where T_i is a very small value if the image is not modified drastically. Thus, the Eq. (2.21) will classify an image as a cover or a stego one.

$$\text{ImgClass} = \begin{cases} Cover & if\ d_i < T_i\ for\ i = 1, 2, ..., 9 \\ Stego & Otherwise \end{cases} \quad (2.21)$$

For the different quality factors, the minimum value of T_2, i.e. for the digit 2, is listed in [1]. It is 3 for the compression factor of 75%. Following the goodness-of-fit table in [1], the static parameters N, q and s of the Eq. (2.19) and the threshold T_2 are initialized to 1.396, 1.731, 0.3549 and 3, respectively. In the experiment, the computed statistics regarding the digit 2 are analyzed.

2.8 Image Datasets Used in the Experiments

The reviewed schemes of this chapter and all the proposed models, stated in Chapter 3 to the Chapter 8, are experimented on several typical image datasets, as shown in Table 2.1. The first four image datasets of Table 2.1 are commonly used in the field of image steganography. Next two image datasets of the table are used to test the performance of the schemes in the diverse images. The last dataset of the table includes the images which are very commonly used in the field of image processing.

Table 2.1: Standard image datasets which are used in experimenting the models of this thesis.

Sr.	Names of the image datasets	Quantity	Referred as*
1	USC-SIPI Standard: Signal and Image Processing Institute, University of Southern California, USA [90]	50	Standard
2	USC-SIPI Texture: Signal and Image Processing Institute, University of Southern California, USA [91]	50	Texture
3	CalTech101: Images of category of 101 of Computer Vision Lab of the California Institute of Technology, USA [6]	5000	CalTech
4	BOSS: Bank of Standardized stimuli [4]	500	BOSS
5	Natural: National History Museum Public Library, UK [69]	50	Natural
6	CRISP-Satellite: Center for Remote Imaging, Sensing and Processing, National University of Singapore [14]	50	Satellite
7	Common: Ten commonly used images (Baboon, Boat, Camera Man, Girl Blonde, Lena, Mona Lisa, Plane, Peppers, Plane and Tiffany)	10	Common

*In this thesis, these short names are used to refer to their respective image dataset.

2.9 Summary and Comments

This chapter summarizes the key literature on reversible data hiding schemes, where continuous attempts were made for improving the prediction accuracy or applying better predictor to achieve enhanced embedding capacity and image quality. Empirical analysis reveals that '0' is the highest appeared prediction errors for all the experimented predictors. Without some rare exceptions, the second highest frequency error is '-1'. The frequency of the

prediction error decreases sharply as the absolute value of the error goes higher. It was also revealed that though the multi-predictor based schemes increase the time complexity by employing multiple predictors these schemes generate more optimal prediction errors and, thus, produces more quantity of embeddable errors. The block median reference predictor also demonstrates better performance in the smaller sized block. The gradient and edge detection based predictors show better accuracy than the block center reference predictors and the neighbor reference predictors. The embedding capacity and the image quality vary in both the SL and the ML schemes with the variation in the prediction accuracy. Although the ML schemes increase the embedding payloads, these destroy the image quality, notably for a larger value of embedding layer. The SL schemes are employed to satisfy small embedding capacity due to their lower complexities. Intentionally image distortion based schemes are used when the cover image itself is a secret of the communication system. None of the reviewed image distortion based schemes uses a predictor in their embedding process.

As an image quality measurement parameter, MSE and PSNR are more famous from the perspective of their uses in the literature. These provide the statistics of temped pixels. In the recent literature, SSIM is taking the leads on the others as this provides a structural comparison between two images and the human visual system is more sensitive to the structural changes in the image. The values of PSNR and SSIM are investigated in the thesis to compare the quality of the stego image with the cover image.

The LSB replacement is detectable by the Chi-Square value. Histogram of adjacent pixel differences is useful in analyzing the resistance of the scheme against attacks as this provides both statistical information and graphical view. Generalized Benford's law is the latest analyzer that provides better information on the changes of pixel values in the stego image. Standard deviations in percentage and the relative entropy are used to test the statistical changes in the images.

Chapter 3

Multi-Block Center Reference Predictor

The existing multi-block center reference predictor based data embedment scheme first divides the cover image into blocks of 3×3 pixels and measures the weighted average of numerous contextual block centers to predict each pixel. Findings show that while predicting the corner pixels of a block, the scheme did not utilize the weights rationally. To rationalize the weighting factor of each context pixel in the prediction rule, in this chapter, a new data embedment scheme is proposed by generating weights for a pixel as an inverse of its air distance from the predicting pixel. The data embedding capacity of the scheme has been further improved by embedding message bits into the errors of a single layer for multiple times. This scheme has generalized the block size from 3×3 pixels to $d\times d$ pixels, where $d\geq 3$ is a common divisor of an image's height and width. The scheme has also modified the existing block skipping criteria to overcome the problem of detecting sharp hill and deep valley around the working block. Each proposal from these bundle has enriched the capability of the proposed scheme to achieve a higher embedding capacity and better stego image quality.

3.1 Introduction

In the block center reference based predictors, the cover image is partitioned into fixed sized blocks. These predictors are of two types–single block center reference (SBCR) based predictor and multi-block center reference (MBCR) based predictor. In the SBCR predictor, the cover image is divided into $m\times m$ sized blocks. All the block pixels are predicted by the value of its block center. The policy differs in MBCR predictor based schemes. In MBCR predictor, the cover image is divided into 3×3 non-overlapping blocks. The center pixel of the working block is termed as the basic pixel while the other pixels in this block are addressed as the non-basic pixels. The neighbor blocks are defined by 4-connectivity rules and thus, each block has four neighbor blocks at its immediate top, right, bottom and left positions. The center pixels of these four neighbor blocks are phrased as the satellite pixels regarding the

working block. When the predictor works in an i^{th} block, the basic is read by C_i and the top, right, bottom and left satellite pixels are read by C_i^T, C_i^R, C_i^B and C_i^L, respectively. The eight non-basic pixels are represented by $b_{i,j}$, where $1 \leq j \leq 8$. In Figure 3.1, white, black and gray colors are used to represent basic, satellite and non-basic pixels respectively. All the non-basic pixels, $b_{i,j}$ are predicted by two or three of C_i, C_i^T, C_i^R, C_i^B and C_i^L in MBCR predictor based schemes. Let the predicted values of $b_{i,j}$ are $P_{i,j}$. The prediction errors $e_{i,j}$ are computed by subtracting the predicted values from the corresponding non-basic pixels, i.e., by $e_{i,j} = b_{i,j} - P_{i,j}$.

These prediction errors are used as an embedding space. The single layer (SL) embedding schemes implant data bits into two most appeared errors, e.g., into the errors of -1 and 0. The performance of these embedding schemes mostly depends on the quantity of these two embeddable errors because both the embedding capacity and the image quality thrive for the improved frequencies of these two embeddable errors. The frequencies of these two errors depend on the accuracy of the applied predictor.

In addition to enhancing the number of embeddable errors, the schemes apply some other strategies to improve the stego image quality. The very commonly used policy is to skip the less embeddable blocks as unchanged. The quantities of embeddable errors vary from block to block. The higher embeddable blocks are termed as the smooth, candidate or embeddable block. On the other hand, the less probable blocks are termed as the complex or less embeddable block. A threshold is applied to the block classification rule to categorize a block as either a smooth or a complex block. To protect the rate of distortions, the schemes skip less embeddable blocks because these blocks conceive fewer bits, but incur much devastation on the image quality of the block.

In this chapter, we propose a novel reversible SL data-hiding scheme resolving the addressed problems stated from the outset of this chapter. In predicting the pixel values, the algorithm first divides the block pixels into two groups–baseline and non-baseline pixels. While predicting the baseline pixels, the scheme computes Euclidean distances of the working pixel from its own block center and the nearest neighbor satellite pixel. For predicting the non-baseline pixels, the scheme computes the Euclidean distances of the working pixel from its basic pixel and two nearest satellite pixels. The relative similarity of a pixel value with it's associated basic and satellite pixels depends on how much the working pixel is near to these

associated basic and satellite pixels. That is why the inverse of the Euclidean distance is used as a weighting factor of the corresponding associated pixel to rationalize the weights in the prediction rules. The proposed scheme further applies more weights for the basic pixel in the prediction rule to devote more honor to the properties of the locality and the proximity. The predicting algorithm generalizes the proposed scheme to be applicable for any size of blocks. To the best of our knowledge, this block generalization strategy is proposed for the first time in the arena of MBCR predictor. The block classification rule is also modified to ensure that no candidate block is to be skipped from being bit-embedded without sacrificing the peak signal to noise ratio (PSNR). Furthermore, the proposed scheme adopts a technique of zero-bit maximization in the message stream to increase the stego image quality.

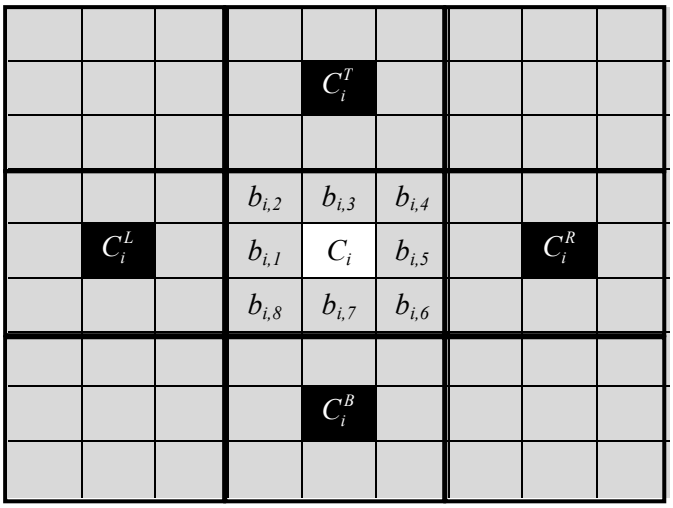


Figure 3.1: Basic, satellite and non-basic pixels in an image.

The rest of this chapter is organized into several sections. The section 3.2 will present the related schemes in a concise manner. The proposed scheme is described in section 3.3. It is investigated that the pixel values of a block are closer to their basic pixel than the satellite pixels. The affair of that block bias tendency is utilized in section 3.4 to further improve the prediction accuracy. The process of multi times data embedment into the errors of a single layer, e.g., into the errors of values of -1 and 0, is proposed in section 3.5. The resistance against statistical attacks is delineated in section 3.6 for several renowned steganalyzers. Finally, some concluding remarks are given in section 3.7.

3.2 Block Center Reference Based Prediction Schemes

The MBCR based predictor is an improved version of the SBCR based predictor. In this section, the prediction methods of an SBCR based prediction scheme, namely Tsai *et al.* in 2009 [88] and two MBCR based prediction policies, namely, Hong *et al.* in 2010 [32] and Lu *et al.* in 2014 [59] are reviewed. These schemes are shortly described in Chapter 2. In this section, their limitations are highlighted.

3.2.1 Predicting Block Contents by Their Own Center Pixel

In 2009, Tsai *et. al.* [88] has proposed a lossless steganographic scheme where the cover image is partitioned into 3×3 sized non-overlapping blocks. In this scheme, the prediction errors are measured by considering the center pixel of a block as a predictive value for the other pixels in the block. One highest appeared positive valued prediction error and another highest appeared negative valued prediction error are chosen to conceal the secret message bits. The scheme performs very poorly in texture images because it predicts very inaccurately in high gradient regions. As no optimization technique is applied to increase the frequencies of these two errors, the embedding capacity is lower. Embedding into fewer pixels and shifting the others by the embedment rules result in lowering the visual quality of the stego image.

3.2.2 Predicting Block Pixels by Associating Multiple Block Centers

In 2010, Hong *et al.*[32] has modified the scheme presented in [88] by predicting the values of all non-basic pixels from its basic pixel and four satellite pixels. They predicted the corner pixels of a block, i.e., $b_{i,2}$, $b_{i,4}$, $b_{i,6}$, $b_{i,8}$ as shown in Figure 3.1 from the weighted average of their two nearest satellite pixels and the basic pixel. For example, $b_{i,4}$ is predicted by using the formula of $P_{i,4}=(C_i+C_i^T+C_i^R)/3$. The other non-basic pixels, i.e., non-corner pixels, are predicted by taking the weighted average of a nearest single satellite pixel along with the basic pixel. For an instance, $b_{i,5}$ is predicted by using the formula of $P_{i,5}=(2C_i+C_i^R)/3$. While predicting the corner pixels, the scheme ignores the neighborhood property because though the basic pixel, C_i is close to the $b_{i,j}$, the C_i is weighted equally in the formula. For example, in the Figure 3.1, air distance, known as Euclidean distance, between C_i and $b_{i,4}$, C_i^T and $b_{i,4}$, and C_i^R and $b_{i,4}$ are 1, 2.24 and 2.24 respectively (the Euclidean distance is measured by

$\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$, where ($x_1$, y_1) and (x_2, y_2) are the positions of two different pixels). Nevertheless, in the predicting equation C_i, C_i^T and C_i^R are weighted equally. Consequently, the scheme predicts these pixels inaccurately.

The scheme classifies the blocks into smooth and complex ones for reducing the computational complexity and image distortions. For performing the classification of the block, first a variance, var(C_i), of the five values, (one basic pixel and four satellite pixels), are computed as $\frac{1}{5}\left\{(C_i-C_m)^2+(C_i^T-C_m)^2+(C_i^R-C_m)^2+(C_i^B-C_m)^2+(C_i^L-C_m)^2\right\}$, where C_m is the mean value of five pixels of C_i, C_i^T, C_i^R, C_i^B and C_i^L. This variance value is compared with a predefined threshold value to classify the block as a smooth or complex. When a satellite pixel is highly unlike compared to others, this variance value becomes more biased by that single satellite pixel. This variance value cannot detect the sharp hill and deep valley as well in and around the block. In addition, no data bits are implanted into the basic pixels and these remain unchanged for the convenience of message extraction and cover reconstruction by the receiver. These basic pixels are 1/8th of the whole image in quantity. In 2014, Lu *et al.* [59] separated all the basic pixels in an imaginary layer upon the stego image. They applied the same prediction and embedment process on these upper layer pixels. Considering the last upper layer as a stego image, the process of generating another upper layer is repeated to embed more bits. The scheme does not provide any new prediction policy; rather, shows only a way of embedding into basic pixels. The correlations among the collected pixels in the upper layer decrease as these pixels come from distinct places. Therefore, the prediction accuracy of the applied predictor decreases at the upper layers.

3.3 Generalized Block-Size Based Predictive Error Embedding Scheme

In this section, the proposed block-size generalization process modified the block classification strategy as well as the data embedment and extraction rules as outlined below. The block generalization scheme presented here provides the flexibility of using blocks of an arbitrary size rather than blocks of 3×3 as used in all previous satellite-pixel based block prediction methods. New block classifier allows more blocks to conceive bits.

3.3.1 Predicting Pixel Values in Arbitrary Sized Blocks

The proposed scheme first divides the cover image into two parts– the upper part *A* and the lower part *I*. Data embedment is done in A by replacing the least significant bits (LSBs) to hide all the assistant information, i.e., a set of values which assist the receiver in extracting message and reconstructing the cover image. The generalized block size based pixel value prediction algorithm, proposed in this chapter, divides the image part *I* into $t_1 \times t_2$ blocks of $d \times d$ size each, where d = {x | x is a natural odd number} and both height and width of *I* is divisible by d, t_1=(height of *I*)/d and t_2= (width of *I*)/d. Each block, $B_{r,c}$ $(1 \le r \le t_1,\ 1 \le c \le t_2)$ has a basic pixel at (i, j) with value $I_{i,j}$ and four satellite pixels at $(i \pm d, j \pm d)$ with values $I_{i\pm d,j}$, $I_{i,j\pm d}$, where $i = (r-1) \times d + d/2$ and $j = (c-1) \times d + d/2$ as depicted in Figure 3.3. The pixel values of the block are accessed by $I_{m,n}$, where $(i - \lfloor d/2 \rfloor) \le m \le (i + \lfloor d/2 \rfloor)$, $(j - \lfloor d/2 \rfloor) \le n \le (j + \lfloor d/2 \rfloor)$. For the space constraints and unknown dimension of d, the satellite pixels at $(i \pm d, j \pm d)$ in Figure 3.3 are exposed as closer to the predictive block rather than showing the actual comparative distance, which is $\lceil d/2 \rceil$ pixels apart from the boundary line of the predictive block.

The values of the pixels $I_{m,n}, (m \ne i, n \ne j)$ in a block are predicted using the weighted average of the basic pixel value $I_{i,j}$ and one or two of the satellite pixel values, $I_{i\pm d,j}$, $I_{i,j\pm d}$. Euclidean distance rules are applied in measuring the weights so that the locality and the proximity are considered rationally. The Euclidean distance between two pixels is the air distance between them and is measured by $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$, where (x_1, y_1) and (x_2, y_2) are the positions of two different pixels. The base-line pixels are predicted from the weighted average of the basic pixel and the nearest satellite pixel, whereas all the other pixels in the block are predicted from the weighted average of the basic pixel and that of its two nearest satellite pixels. Let $I_{u,v}$ and $I_{x,y}$ are the two nearest satellite pixels while predicting a non-base-line pixel $I_{m,n}$. For example, $I_{u,v} = C_i^T$ and $I_{x,y} = C_i^R$ in Figure 3.1 when $I_{i,j} = C_i$ and $I_{m,n} = b_{i,4}$. Say, the Euclidean distances of $I_{m,n}$ from $I_{i,j}$, $I_{u,v}$ and $I_{x,y}$ are $D_{m,n}^{i,j}$, $D_{m,n}^{u,v}$ and $D_{m,n}^{x,y}$ respectively. Again, let the nearest satellite pixel is $I_{x,y}$ when $I_{m,n}$ is a base-line residual, e.g., $I_{x,y} = C_i^R$ when $I_{m,n} = b_{i,5}$. The Euclidean distance between $I_{m,n}$ and each of $I_{i,j}$ and $I_{x,y}$ are

$D_{m,n}^{i,j}$ and $D_{m,n}^{x,y}$ respectively. Let the predicted value of a pixel $I_{m,n}$ is $P_{m,n}$. The prediction of a base-line pixel $I_{m,n}$ is performed by using Eq. (3.1). The expression is simplified in the Eq. (3.2). In the same way, the non-base-line pixels are predicted by using Eq. (3.3).

$$P_{m,n} = \frac{\frac{1}{D_{m,n}^{i,j}} \times I_{i,j} + \frac{1}{D_{m,n}^{x,y}} \times I_{x,y}}{\frac{1}{D_{m,n}^{i,j}} + \frac{1}{D_{m,n}^{x,y}}} \tag{3.1}$$

$$P_{m,n} = \frac{D_{m,n}^{x,y} \times I_{i,j} + D_{m,n}^{i,j} \times I_{x,y}}{D_{m,n}^{i,j} + D_{m,n}^{x,y}} \tag{3.2}$$

$$P_{m,n} = \frac{D_{m,n}^{u,v} \times D_{m,n}^{x,y} \times I_{i,j} + D_{m,n}^{i,j} \times D_{m,n}^{x,y} \times I_{u,v} + D_{m,n}^{i,j} \times D_{m,n}^{u,v} \times I_{x,y}}{D_{m,n}^{i,j} \times D_{m,n}^{u,v} + D_{m,n}^{u,v} \times D_{m,n}^{x,y} + D_{m,n}^{i,j} \times D_{m,n}^{x,y}} \tag{3.3}$$

The values of (*u, v*) and (*x, y*) are distinct and indicate the location of two different satellite pixels. Therefore, based on the location of (*m, n*) in the block, the values of (*u, v*) and (*x, y*) are selected from $(i \pm d, j \pm d)$. Considering clockwise direction, the values in the first quadrant are $(i, j-d)$ and $(i+d, j)$, whereas these are $(i+d, j)$ and $(i, j+d)$ in the second quadrant, $(i, j+d)$ and $(i-d, j)$ in the third quadrant and $(i-d, j)$ and $(i, j-d)$ in the fourth quadrant. All the basic pixels remain unchanged for the convenient of predicting the same $P_{m,n}$ at the receiver end. The prediction errors $E_{i,j}$ are then calculated by using the Eq. (3.4).

$$E_{i,j} = I_{i,j} - P_{i,j} \tag{3.4}$$

Two highest appeared errors H_p and H_n are recorded, where $H_p > H_n$, rather than one positive and one negative valued error, as considered in [32], because in many images both H_p and H_n are greater than or equal to zero, as shown in Figure 3.2. This process will improve the embedding capacity a bit.

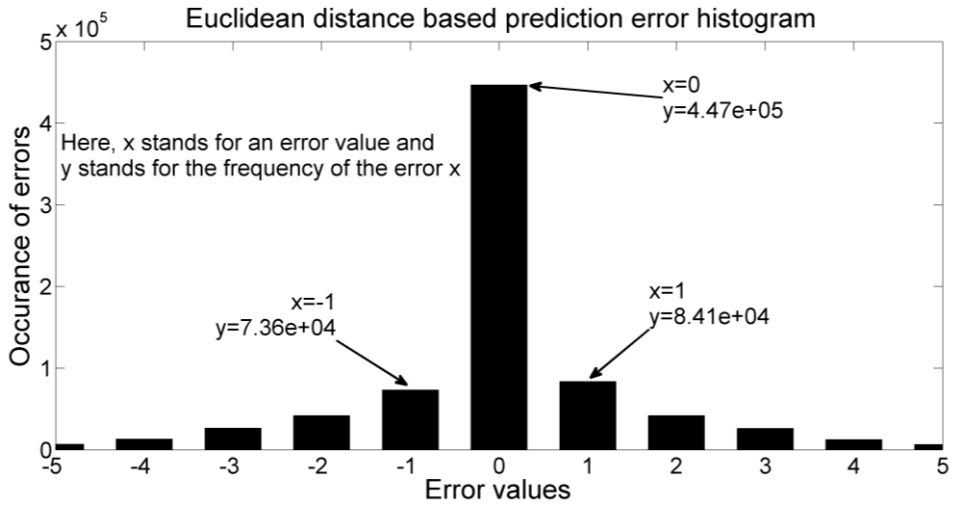


Figure 3.2: Prediction error histogram where both two highest peaked errors are positive valued.

3.3.2 Block Classification Using Block Variance

To improve the image quality, the scheme in [32] avoids less embeddable blocks. The scheme first estimates block whether it contains a large number of embeddable pixels or not. The estimation is performed by measuring the block variance. The variance value is computed from one basic pixel and four satellite pixels. In their equation, the variance value is largely influenced by a single unlike satellite pixel even if the block contains many embeddable pixels. For example, if a satellite pixel exists in a sudden transitional area, e.g., in a texture region, this pixel value will influence the classification rules to mark the block as a non-smoothed block even if the value of the other three satellite pixels and the block's own pixels are very close to each other. The block skipping rules cannot detect a sharp hill or a deep valley in a small vicinity as well. In a deep valley or in a sharp hilly region, the values of block pixels differ largely from the satellite pixels; however, the four satellite pixels will influence the classifier to consider the block as a smoothed one. The predictor will then predict in that block very inaccurately. To protect the misguidance by the block classifier stated in [32], the block variance calculation process of [32] is modified in the following ways. First, a mean of the four satellite pixels (MoS), is measured. For all natural images, the MoS is very close to $I_{i,j}$, i.e., MoS $\approx I_{i,j}$. The variance of (MoS, $I_{i,j}$) is applied to grade a block as smooth or complex by comparing it with a predefined threshold, TH, as depicted in Eq. (3.5) through Eq. (3.7).

$$MoS = round\left(\frac{I_{i-d,j},+I_{i,j-d}+I_{i+d,j}+I_{i,j+d}}{4}\right) \tag{3.5}$$

$$Var_{r,c} = \left(I_{i,j} - MoS\right)^2 \tag{3.6}$$

$$BS_{r,c} = \begin{cases} complex & if\ \ Var_{r,c} > TH \\ smooth & Otherwise \end{cases} \tag{3.7}$$

The MoS computed using the Eq. (3.5) reduces the effect of a single unlike satellite pixel and the variance computed using the Eq. (3.6) detects the existence of any sharp hill or deep valley around the basic pixel.

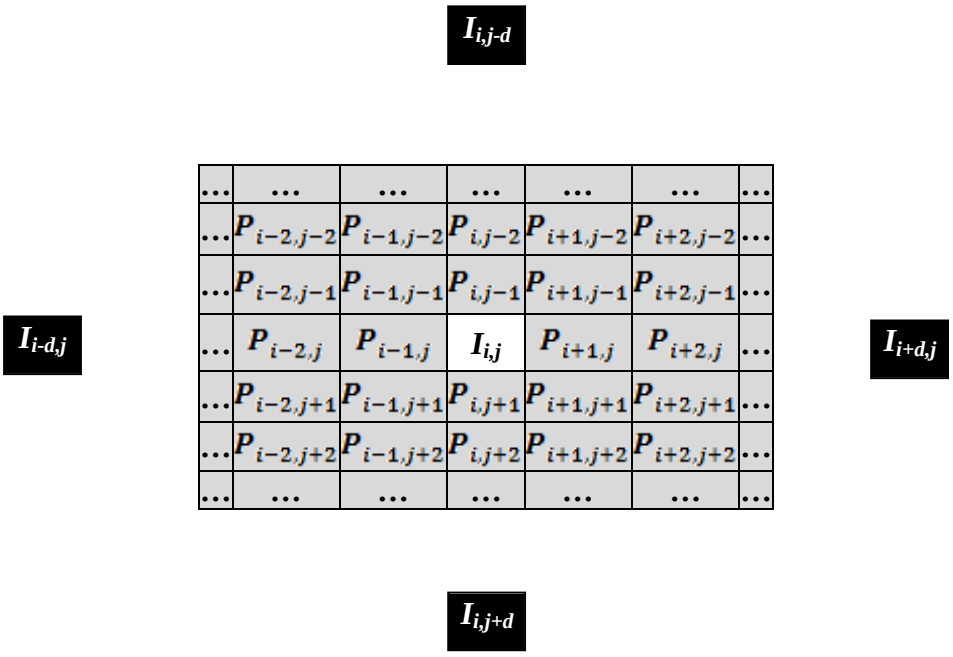


Figure 3.3: Prediction of pixels in the generalized block

3.3.3 Embedment Procedure Used by the Data Hider

The method of [32] first divides the cover image into two parts A and I, where the A comes from the upper portion of the cover image and the I comprises of the remaining portion of the cover. The process of managing the assistant information, say AI_{info}, and hiding it in the image part A by using the LSB replacement method are explained in [32]. Before replacing the LSBs of A, these LSBs of A are extracted and concatenated with the secret message, known as processed secrets. Next, the bit stream of AI_{info} is implanted into the pixels of A by the LSB replacement method. Let the [32] generated stego part of A is $\tilde{A}$. The scheme implants the processed secrets into the other part I by using the prediction error based process. In the

prediction error based process, a predictor estimates the pixel values of all the block in *I*. It, then, computes the prediction errors $E_{i,j}$ by subtracting the predicted values $P_{i,j}$ from their respective cover values $I_{i,j}$. The bit stream of the processed secrets is implanted into these computed error values $E_{i,j}$ by using the Eq (3.8). The implantation process modifies the prediction errors. Let the modified stego errors are $\tilde{E}_{i,j}$. Finally, the scheme forms the stego values of the lower part, $\tilde{I}_{i,j}$ of each (i, j) location by adding each of the $\tilde{E}_{i,j}$ with their corresponding predicted values $P_{i,j}$. The scheme forms the stego image SI by concatenating $\tilde{I}$ with $\tilde{A}$.

$$\tilde{E}_{i,j} = \begin{cases} E_{i,j} \odot s & if \quad E_{i,j} = H_p \quad or \quad E_{i,j} = H_n \\ E_{i,j} \odot 1 & if \quad E_{i,j} > H_p \quad or \quad E_{i,j} < H_n \\ E_{i,j} & Otherwise \end{cases} \tag{3.8}$$

Where $\odot$='-' if $E_{i,j} \leq H_n$ or $\odot$='+' if $E_{i,j} \geq H_p$, s is a message bit and $|.|$ is used to represent the absolute value.

3.3.4 The Extraction Procedure Used by the Receiver

At the receiver, the stego image *SI* is partitioned into two parts $\tilde{A}$ and $\tilde{I}$. From the LSBs of $\tilde{A}$, the assistant information AI_{info} is reproduced. The block classification threshold *TH*, the length of the message, two highest appeared errors H_p and H_n are separated from the AI_{info} . As the basic pixels were remained unchanged, the same predicted values $P_{i,j}$ are generated for each pixel of a specific block $B_{r,c}$ by using the Eq (3.2) and (3.3). The stego prediction errors $\tilde{E}_{i,j}$ are obtained as $\tilde{I}_{i,j} - P_{i,j}$. Each message bit s is extracted by using the Eq (3.9). Then the original cover prediction errors $E_{i,j}$ are calculated by using the Eq (3.10).

$$s = \begin{cases} 0 & if \quad \tilde{E}_{i,j} = H_p \quad or \quad \tilde{E}_{i,j} = H_n \\ 1 & if \quad \tilde{E}_{i,j} = H_p + 1 \quad or \quad \tilde{E}_{i,j} = H_n - 1 \end{cases} \tag{3.9}$$

$$E_{i,j} = \begin{cases} \tilde{E}_{i,j} \odot 1 & if \quad \tilde{E}_{i,j} > H_p \quad or \quad \tilde{E}_{i,j} < H_n \\ \tilde{E}_{i,j} & Otherwise \end{cases} \tag{3.10}$$

The extracted message contains the LSBs of A as these LSBs were concatenated with the message before the data embedment by the data hider. The upper part A of the image is

reproduced from the stego image part $\tilde{A}$ by replacing the LSBs. The lower part of the cover image $I_{i,j}$ is reconstructed as $P_{i,j} + E_{i,j}$. Finally, concatenating A and I reproduces the cover image CI.

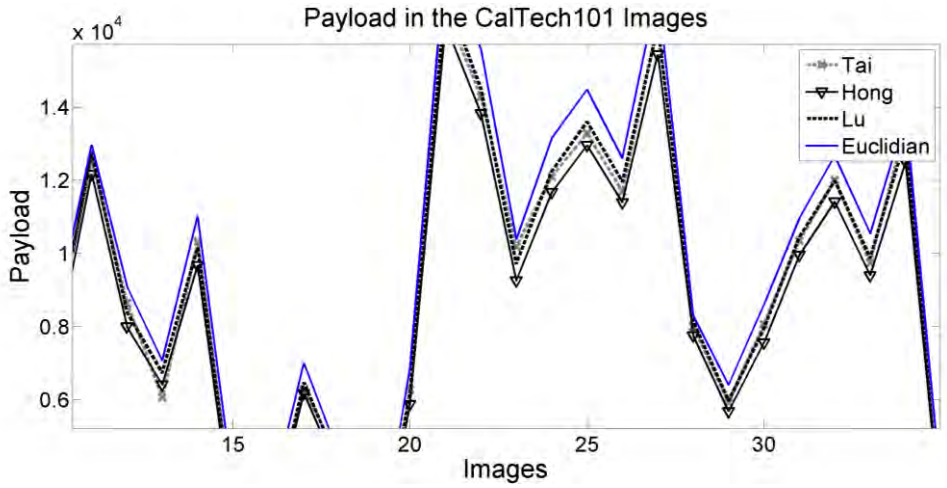


Figure 3.4: Payloads achieved in various images by the different schemes.

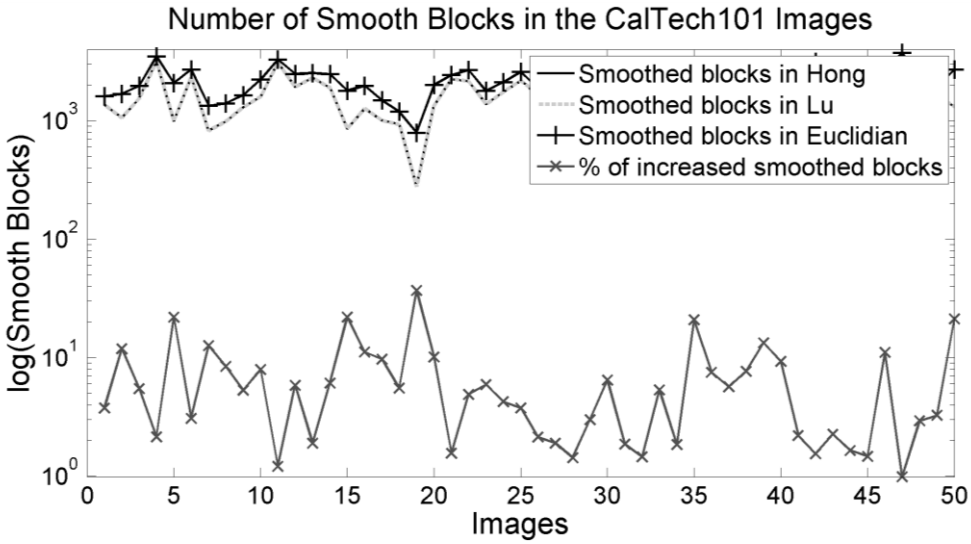


Figure 3.5: Performance of new block classifier.

3.3.5 Result Analysis

In the experiments, we use the image datasets of BOSS, Caltech, Texture, Satellite, Natural, Standard and Common, as stated in Table 1.1 of Chapter 1. The schemes of Tai *et al.*[83],

Hong *et al.*[32] and Lu *et al.*[59] are experimented and compared with our proposed scheme. The proposed scheme dominates the others in all the measuring performance parameters. The outcomes of the experimental results are demonstrated below.

3.3.5.1 Payload comparison

Figure 3.4 clearly depicts the superiority of the proposed scheme in achieving higher embedding payloads. It happens for two reasons - the predictor proposed in this scheme predicts more accurately than the other scheme and the new block classifier classifies blocks more rationally. The new classifier allows more blocks to participate in conceiving message bits. The performance of the classifier is delineated in Figure 3.5 in log scale. This figure shows that the number of smooth blocks categorized by the proposed scheme is more than the scheme proposed by Hong *et al.* in 2010 [32]. The percentage of improvement in the number of smooth blocks is always positive, which indicates that the proposed classifier grades more blocks as a smooth block for the same level of the threshold value.

3.3.5.2 Comparison of Stego Image Quality

The stego image quality is compared in terms of the values for both the peak signal to noise ratio (PSNR) and the structural dissimilarity index (SDIM). The amount of PSNR loss due to data implantation is measured. The quantities of payloads per unit PSNR_Loss achieved in the different images of CalTech dataset are demonstrated in Figure 3.6. The figure states that the proposed scheme enhances the number of embedded bits for each 1dBm loss of PSNR values, i.e., the ratio between the number of implanted bits and the amount of PSNR loss is higher in the proposed scheme. Similarly, the Figure 3.7 depicts that the proposed scheme embeds more data for the same amount of SDIM value. This investigation concludes that, for the same level of image distortions, the proposed scheme embeds more data bits into an image.

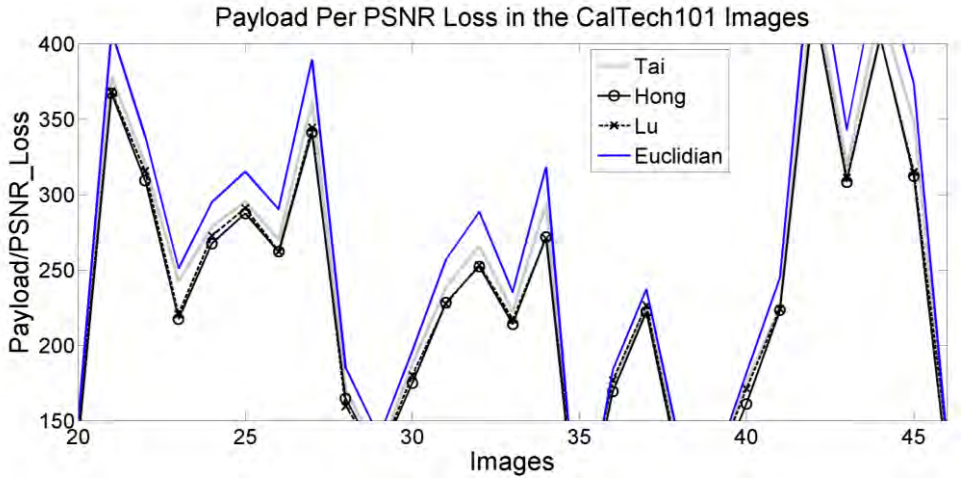


Figure 3.6: Payloads per unit PSNR loss achieved in various images by the different schemes.

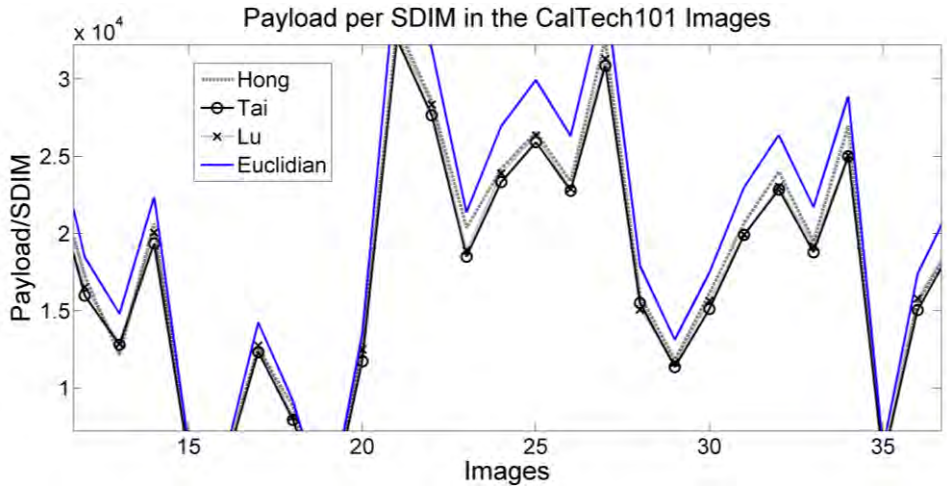


Figure 3.7: Payloads per SSIM achieved in various images by the different schemes

3.3.5.3 Comparison of Payloads in Diverse Image Dataset

To study the impacts of the proposed scheme on various types of images, experiments are conducted in several standard image data sets. The achieved payloads in the image datasets of BOSS and Satellite are demonstrated in Figure 3.8 and Figure 3.9, respectively. These two figures clearly show that the proposed scheme outperformed over all the schemes for different types of image of data sets.

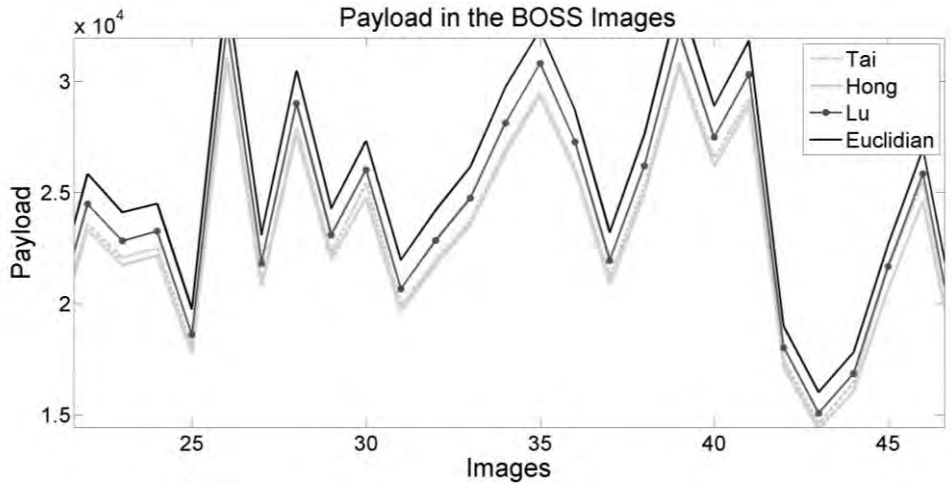


Figure 3.8: Payloads of the different schemes for the BOSS image dataset.

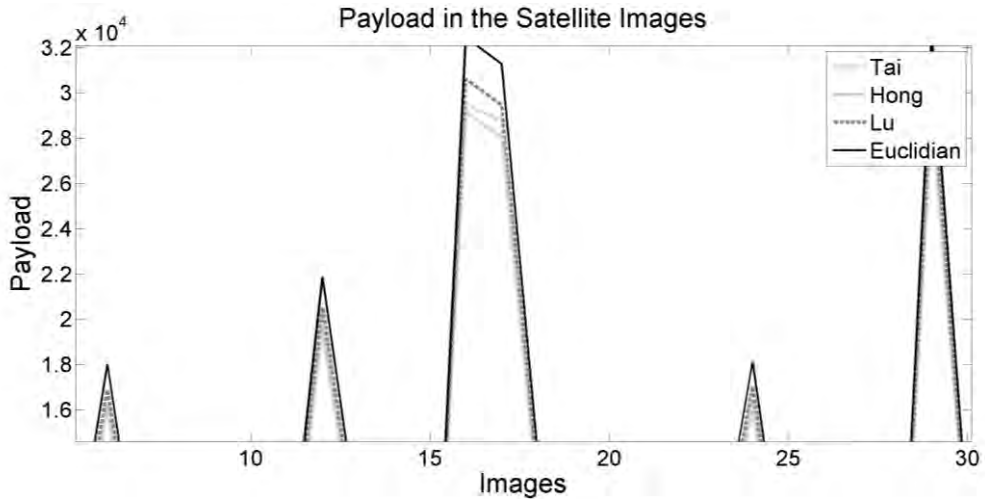


Figure 3.9: Payloads of the schemes in the Satellite image dataset.

3.4 Block Biasness Tendency Affairs for Further Improving the Prediction Accuracy

Investigation shows that the pixels of a block are more similar to their block center, i.e., basic pixel. That is why, in predicting a pixel $I_{i,j}$, the proposed predictor applies the weighting factor of the context pixels as the inverse of their Euclidian distances from the predicting pixel. In natural images, both smooth and greater transitions of intensities among the pixel values, known as image gradient, are happened in a frequent manner. The value of the gradient becomes large when a big transition is observed. In a place of the greater gradient, two pixels

become non-correlated when pixels come from a bit apart places. On the contrary, the amount of gradient decreases sharply for two pixels of closed positions. In a block, the block pixels are closer to the basic pixel than their satellite pixels. Hence, further augmenting the assigned weights to basic pixel, in the Euclidian distance based weighted average predictor, will improve the prediction accuracy. The Eqs (3.2) and (3.3) for the prediction rules are modified in Eqs. (3.11) and (3.12), respectively, by incorporating a new parameter α. In Eqs (3.11) and (3.12), the basic pixel is given more weight by assigning a value to α by greater than one. The α is restricted not to be greater than 3 because for a big value of α this predictor will be biased to estimate the basic pixel value directly and in that case, the objective of a weighted average of multi-block centers will be demolished. Besides, thereafter, the performance of the predictor decreases, as shown in Figure 3.10. This parameter α is used to meet the pixels' block bias tendency.

$$P_{m,n} = \frac{\alpha \times D_{m,n}^{x,y} \times I_{i,j} + D_{m,n}^{i,j} \times I_{x,y}}{\alpha \times D_{m,n}^{i,j} + D_{m,n}^{x,y}} \tag{3.11}$$

$$P_{m,n} = \frac{\alpha \times D_{m,n}^{u,v} \times D_{m,n}^{x,y} \times I_{i,j} + D_{m,n}^{i,j} \times D_{m,n}^{x,y} \times I_{u,v} + D_{m,n}^{i,j} \times D_{m,n}^{u,v} \times I_{x,y}}{\alpha \times D_{m,n}^{i,j} \times D_{m,n}^{u,v} + D_{m,n}^{u,v} \times D_{m,n}^{x,y} + D_{m,n}^{i,j} \times D_{m,n}^{x,y}} \tag{3.12}$$

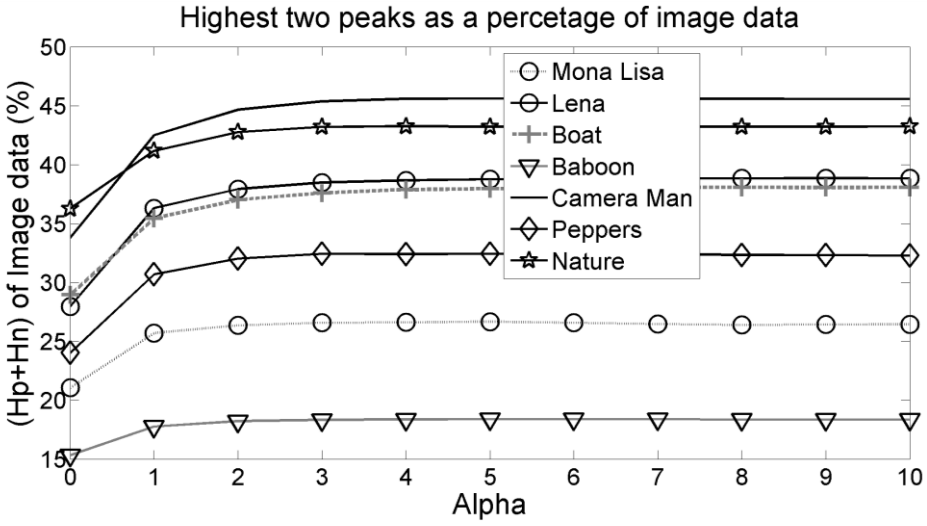


Figure 3.10: The effect of Alpha in estimating pixels.

3.4.1 Result Analysis

To demonstrate the improvement for the utilization of α in the prediction rules, the Figure 3.11 is depicted. The comparison is performed between the Euclidian distance based predictor and the predictor that consider block bias tendency. The embedding capacity is improved by the

uses of the second predictor and hence, the frequencies of the embeddable errors are higher in this predictor. In this experiment the value of α was set to 3 as at α=3, the predictor shows the best performance, as shown in Figure 3.10.

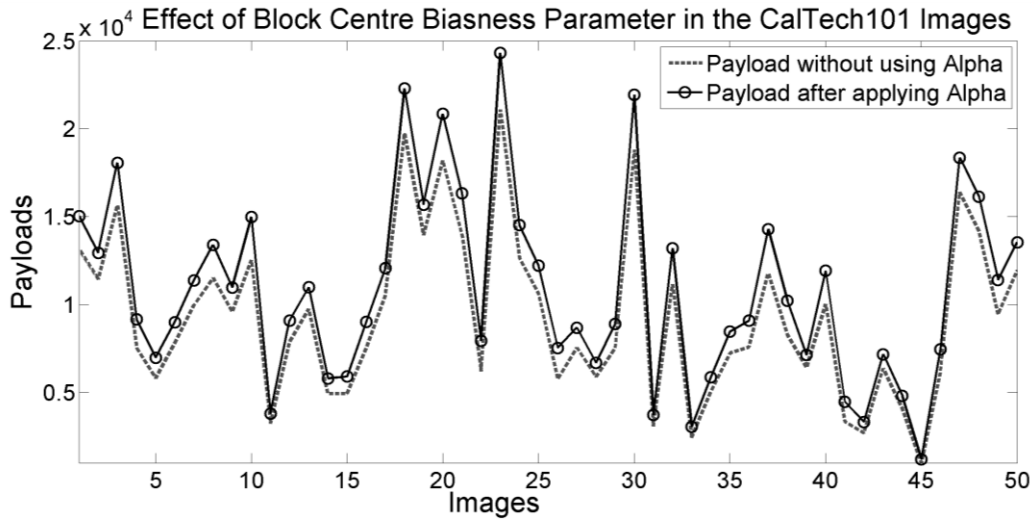


Figure 3.11: Improvement of payloads for α=4.

3.5 Multi-Cycle Embedding Procedure

In this chapter, we implement a multi-cycle embedding procedure where the scheme reuses the stego errors in each repeating step to implant bits using the same embedding rules. Applying the same implantation procedure for multiple times in the stego errors, the scheme enhances the embedding capacity. This multi-cycle embedment process is not deprecated only when the payload is big, but the data should be implanted into the same image; however, this process may reduce the image quality notably and the location map can disallow many pixels to accept bits. The location map is a binary map that keeps track of these pixels, which may exceed the grayscale if data bits are implanted into these. Therefore, 0 and 255 valued pixels are not used in the single cycle data embedment procedure. Similarly, {0, 1, 254, 255} and {0, 1, 2, 253, 254, 255} valued pixels are not employed for embedding bits in the 2-cycle and 3-cycle data embedment procedure. A binary location map tracks the positions of these pixels in the image. This location map is compressed and implanted into the image along with other data. However, if the pixels within the range of [0, k] and [255-k, 255] are not used in *k*-cycle data embedment procedure and if these pixels are large in quantity, the embedding space will be shrunk dramatically. The investigation reveals that 34%, 0%, 14%, 64%, 38% and 20% images

in the dataset of Natural, BOSS, CalTech, Standard, Texture and Satellite, respectively do not contain any pixel with the values of 0 or 255. The investigation also reveals that 30%, 0%, 14%, 64%, 36% and 16% images of the respective standard dataset images do not contain any pixel with the values of {0, 1, 254, 255}; while 28%, 0%, 12%, 60%, 34% and 12% images of the same datasets do not contain any pixel with the values {0, 1, 2, 253, 254, 255}. Therefore, we have lots of images where we can apply up to triple cycle data embedment process without the uses of location map. Even if the generation of location map is mandatory, this map can be handled by the existing methods stated in [32, 57]. The multi-cycle data embedment and data extraction processes are outlined in the following subsections.

3.5.1 Embedment of Secret Information

The embedding procedure takes a bit of information, *s* from the to be implanted stream S_c at each embedding attempt and implants it into an error $E_{i,j}$. Let PTR be a pointer that points to the first position of S_c and returns a message bit *s* on each of its sequential access. After each access, this pointer moves to the next position in the bit stream. Two steps as depicted below do the multi-times data embedment:

Step 1. Each message bit *s,* which is returned by the *PTR* is embedded in the prediction error $E_{i,j}$ by using the Eq (3.7).

Step 2. If $PTR \neq NULL$, i.e., all the blocks are tried, then all the stego prediction errors $\tilde{E}_{i,j}$ produced by using the Eq (3.7) are assigned to $E_{i,j}$, i.e. $E_{i,j} = \tilde{E}_{i,j}$, and Step 1 is repeated.

The scheme counts the number of bits, say *l*, that is implanted in its last cycle. This *l* is converted to 16 bits binary. The embedding cycle *k* is converted to 3 bits binary. This 19 bits, i.e., 16 bits+3 bits, are finally implanted as a part of the assistant information by the last cycle in the upper part *A* of the image through substitution of LSBs as depicted in the Section 3.3.3. Say the upper stego part is *A*. The stego errors are added to the corresponding predicted values to form stego lower parts $\tilde{I}$. These two are concatenated to form the final stego image *SI*.

3.5.2 Extraction of Secret Information

As depicted in Section 3.3.1, the stego image *SI* is divided into two parts– the upper part *A* and the lower part . The assistant information is collected from the stego image part . If data embedment for multiple cycles is realized from the assistant information, the hidden message in the last cycle is extracted first by using the Eq. (3.9) and the errors are reconstructed by using the Eq. (310). After completing a cycle of data extraction, all the prediction errors $E_{i,j}$ are assigned to stego prediction errors $\tilde{E}_{i,j}$. The Eqs (3.9) and (3.10) are applied again to $\tilde{E}_{i,j}$ in the next extraction cycle to extract the remaining data and to reconstruct the cover prediction errors. This process is repeated until all the message bits are not extracted and the cover errors are not reconstructed.

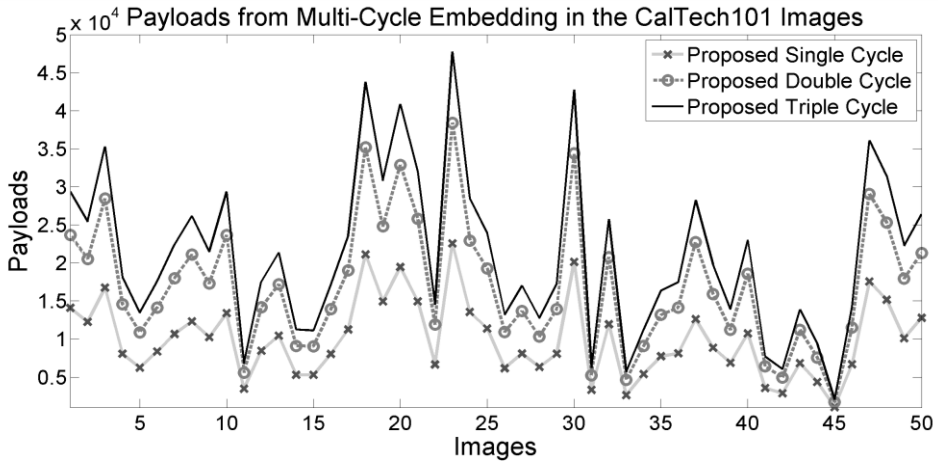


Figure 3.12: Comparison of payloads among the multi-cycle data embedment processes for the CalTech image data set.

3.5.3 Result Analysis

The proposed single cycle data embedment procedure stated in Section 3.3 demonstrates better results than its competing schemes regarding all the performance measuring parameters such as embedding payloads, PSNR. In this section, the multi-cycle processes are compared among themselves. As the stego image quality decreases with the increment of embedding cycles, only the results of first three cycles are presented in the following figures. The Figure 3.12 demonstrates that the payload increases as the embedding cycle goes higher. The rate of improvement from the second cycle to the third cycle is smaller compared with the improvement from the first cycle to the second cycle. After each embedding cycle, the quantity

of embeddable errors decreases. The same scenarios are investigated in the Figure 3.13 and Figure 3.14, where the payload per unit PSNR loss and payload per unit structural similarity index (SSIM) are considered. Thus, it is investigated that the multi-cycle embedment schemes provide higher payloads per level of image distortions.

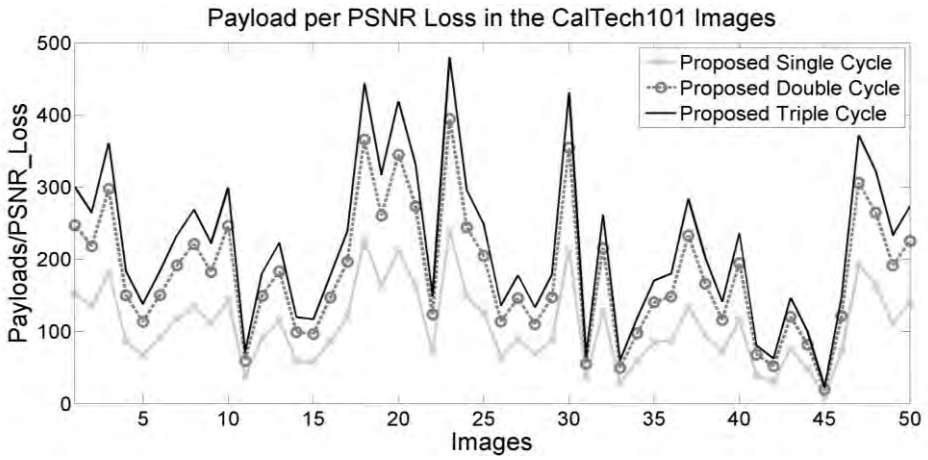


Figure 3.13: Payload per unit PSNR loss in the multi-cycle scheme.

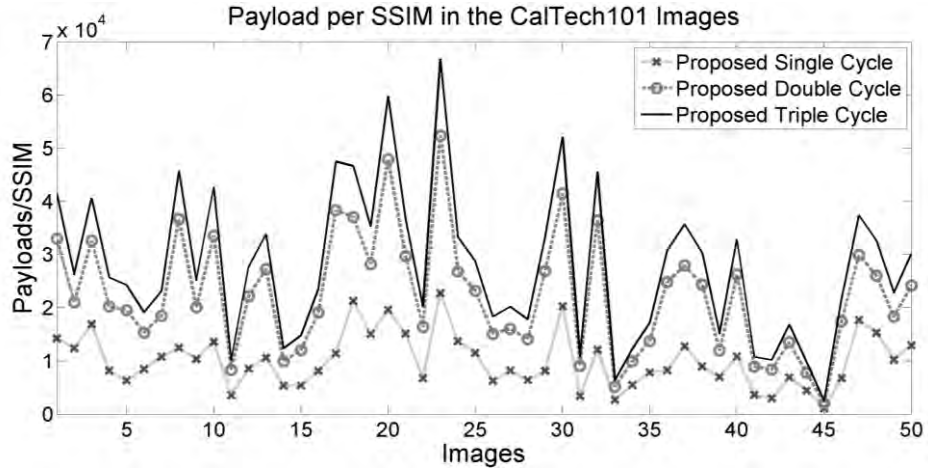


Figure 3.14: Payloads per SSIM in the multi-cycle scheme.

3.6 Resistance to Attacks

The attackers use various steganalyzers to realize the existence of the hidden message in an image and to retrieve the secret in an unauthorized manner. The developers and the researchers also test the resistance of the proposed or implemented model by using different Steganalyzers

creating many virtual attacks. Several Steganalyzers like a histogram difference test (presented in the Section 2.7.2), standard deviations and correlation coefficient test (presented in the Section 2.7.3) and relative entropy test (presented in the Section 2.7.4) are used to analyze the robustness of the proposed scheme. The results obtained are tabulated in Tables 3.1-3.3. The entropy of cover image pixels and stego image pixels are measured first. The difference between these two entropies is presented in the Table 3.1. The table states that the difference values are very small. These smaller values confirm that the relative changes in the stego image with respect to the cover image are undetectable under the statistical analysis.

Table 3.1: Differences in entropy values between the cover and the stego images

Image	Tsai	Hong	Euclidean Alpha	Euclidean Cycle 2	Euclidean Cycle3
Boat	0.00328	0.004079	0.006822414	0.021759	0.034933
Camera Man	0.019866	0.026278	0.023318849	-0.52975	-0.47127
Tiffany	0.009402	0.012604	0.014059786	-0.25278	-0.22789
Lena	0.004386	0.005465	0.005598739	-0.24291	-0.22126
Baboon	0.000509	0.000723	0.002899321	0.017921	0.018189
Peppers	0.003265	0.003755	0.004240759	-0.17741	-0.17509
Plane	0.014237	0.016943	0.015655001	0.030364	0.079689

Table 3.2: The correlation coefficients and percentage of changes in the standard deviations of the cover and the stego images

Tsai		Hong		Euclidean Alpha		Euclidean Cycle 2		Euclidean Cycle3	
Change in Std* (%)	Corr Coeff**	Change in Std* (%)	Corr Coeff**	Change in Std* (%)	Corr Coeff**	Change in Std* (%)	Corr Coeff**	Change in Std* (%)	Corr Coeff**
0.304	1.000	0.295	0.986	0.289	0.986	4.777	0.921	4.784	0.921
0.177	1.000	0.195	0.991	0.187	0.991	6.120	0.934	6.107	0.934
0.534	1.000	0.548	0.989	0.515	0.989	9.583	0.910	9.521	0.909
0.658	1.000	0.657	0.990	0.670	0.990	1.846	0.849	1.867	0.849
0.881	1.000	0.884	0.979	0.892	0.979	0.308	0.857	0.300	0.857
0.133	1.000	0.128	0.992	0.143	0.992	4.524	0.913	4.543	0.912
0.581	1.000	0.588	0.986	0.585	0.986	9.958	0.866	9.971	0.866

*Change in Std: % of Changes between the standard deviations of cover and stego image,

**CorrCoeff: Correlation Coefficients.

Table 3.3: D values (stated in Section 2.7.2) computed from vertical and horizontal difference histogram

Image	Tsai	Hong	Euclidean Alpha	Euclidean Cycle 2	Euclidean Cycle3
Boat	950.20	975.10	903.35	890.45	919.20
Camera Man	1210.37	1214.15	1216.96	836.09	844.25
Tiffany	1278.37	1285.03	1218.76	741.20	797.66
Lena	3261.18	3250.73	3080.94	2861.22	2855.46
Baboon	463.99	471.69	432.85	419.54	418.66
Peppers	1868.70	1876.27	1718.80	1121.48	1172.47
Plane	1507.52	1489.17	1463.34	838.13	930.42

Table 3.2 indicates that the values of the correlation coefficients, which are measured from the cover and the stego images, are much larger, i.e., higher than 0.85. The differences of standard deviations, measured between the cover and the stego images, are always less than 10%. These tabulated values ensure that the cover and the stego images are statistically more alike. These images accept message bits up to 20000 to 60000 depending on different schemes. Nevertheless, the value of D (stated in the Section 2.7.2) is less than 3000 in most of the images in different schemes. These mean that it is very hard to detect the changes in the stego image. Thus, the proposed scheme shows enough resistance against statistical attacks.

3.7 Summary and Comments

In every day, the volume of information communicated is increasing rapidly and thus the requirement for hiding the large volume of data. The proposed scheme improves the embedding capacity by employing several policies. Generalizing the partitioning size in the satellite-based prediction scheme is a unique contribution. The uses of Euclidian distance rules to rationally weight the context basic and satellite pixels and the application of alpha value in the prediction rules to put emphasis on bias tendency of block pixels towards their basic pixel have improved the prediction accuracy significantly. This accuracy in the prediction policy has acted as an agent to improve the frequencies of the two most appeared prediction errors. Consequently, the scheme finds more space to implant bits. The multi-cycle embedment policy is designed to satisfy the demand for larger payloads. Though the embedment of bits for multiple times into the same image will reduce the image quality, the quality degradation is not significant, i.e., remains below a threshold margin level (the value of SSIM is more than 0.85 in more than 50 percent of images) up to three cycles data embedment. These multi-cycle

schemes will be a notable contribution in the field of image steganography. The statistical analyzers show that the scheme bears enough resistance against attacks.

Chapter 4

Hybridizing Multiple Predictors

In this chapter, an optimal prediction error generation method is proposed through employing multiple predictors and selecting the best one or a combination of them. The proposed scheme predicts multiple values for each pixel of the cover image, each of which is generated through the employment of a separate predictor. The scheme then measures the prediction errors for all of them. Investigations show that, for each predictor, the highest frequency appeared prediction error among the computed ones for an image is always 0 and, with some exceptions, the prediction error with the second highest frequency is -1. In the exceptional cases, 1 is observed as the second highest frequency predation error and then the frequencies of -1 and 1 differ slightly. Hence, the single layer-embedding scheme proposed in this chapter implants message bits into the prediction errors of -1 and 0. After generating the prediction errors using each of the predictors, the scheme sequentially explores for the prediction error of -1 or 0 for each pixel among those measured by the different predictors and embeds a message bit into that error. If none of the prediction errors are embeddable, then the scheme applies several linear relations on those computed errors of the pixel to generate more errors, called the hybrid errors, with an intention of generating 0 or -1 as the prediction error for that pixel. The process increases the quantities of the two embeddable errors, and thus, increases the embedding capacity. These improved frequencies of the embeddable errors also reduce the amount of total error shifting and hence, lessen the degradation in the image quality. The experimental results presented in this chapter demonstrate significant improvements in embedding capacity and maintaining the stego-image quality compared with both the multi-predictor based schemes and single predictor based schemes presented in the literature.

4.1 Introduction

The prediction error based schemes predict each cover pixel value by employing one or multiple predictors. The prediction errors are computed by subtracting the predicted values

from the values of the corresponding cover pixels. Few of these prediction errors centered at 0, known as the embeddable errors, are used to implant secret message bits. The number of embeddable errors increases when the prediction process estimates the pixel value more accurately. The more the embeddable errors, the more the embedding capacity as well as the better the stego image quality. Based on this philosophy, the prediction error based schemes, in the state-of-the-art, incorporated diverse prediction methodologies to improve the prediction accuracy. When the prediction accuracy is improved, the prediction errors present a better spiky Laplacian-like distribution [72] with the peak at zero or very close to zero. This spiky Laplacian-like distribution implies that the number of errors in and around the peak presenting error is increased. This is evident that the highest appeared error in the prediction error histogram is always 0. The second highest frequency error in most cases is -1. The value of the second most occurrence error is 1 in very rare cases and if it is, then the frequency of the error value 1 is very close to that of -1. Hence, the single layer (SL) data embedment schemes, e.g., [51], generally recon only -1 and 0 as the members of the embeddable error set.

Though the predictor with better accuracy increases the frequencies of the errors of -1 and 0, these will not greedily maximize the frequencies of these two errors, because a better predictor reduces only the magnitudes of prediction errors by improving the accuracy rather than intending to produce the errors of -1 and 0 only. Consequently, an improvement of prediction accuracy, while predicting a pixel value, does not ensure that the value of the corresponding prediction error will be -1 or 0. Indeed, then the predictor concentrates more prediction errors near 0. The errors rather than -1 and 0 are useless in the SL data embedment process as these are unable to accept any message bit. This chapter presents a new methodology that applies multiple predictors, say n predictors, on each pixel prediction for increasing the possibilities of generating prediction errors of -1 or 0 by using at least one of the predictors. In a case, when none of the predictors generate a prediction error of -1 or 0, an attempt is made to produce these two errors by employing the computed n prediction errors in m different linear relations as the output of them. Thus, each optimal prediction error is extracted from these $n+m$ errors. Simulation results confirm that the proposed scheme provides almost 10%–9233% higher embedding capacity depending on the texture contents of the cover image, while ensuring the improved image quality compared with the competing ones.

The whole process of the proposed scheme is organized into several sections. Section 4.2 provides a description of the prediction policies and categorizes them from different

perspectives. This section is furnished with brief descriptions of the related existing schemes. The proposed multi-predictor based data embedment scheme is outlined in Section 4.3. Section 4.4 is organized to provide a demonstration of the experimental results and a discussion on them. Finally, Section 4.5 is for concluding the chapter.

4.2 Prediction Policies for Generating Rich Embedding Space

Prediction errors are generated in the first phase of the data embedment process, known as the prediction phase. In the prediction phase, the predictor predicts a pixel. The prediction errors are measured by subtracting the predicted values from the corresponding cover pixel values. In Section 2.2, several diverse prediction methodologies are presented for the prediction phases of the data embedment processes. These methodological varieties come from the policies of weighting the associated pixels in the prediction rules, the application of gradient responses, the uses of multi predictors, the exploitation of a number of reference pixels taken as context values in the prediction rules and their positions in the image. All of these predictors predict either a single pixel ([31]) or a group of pixels ([32]) from a set of homogeneous context pixels by exploiting the features of spatial association among the neighboring pixels of the cover image. Predictors are, therefore, classified as either single pixel predictor (SPP) or block pixel predictor (BPP). Associated pixels, used to predict the target ones by the predictor, remain unchanged during the data hiding process for the benefits of extracting the secret messages at the receiver end. The research presented in this chapter aims to enhance both the embedding capacity and the stego image quality by increasing the quantities of embeddable errors. The competing schemes in the area of SPP and BPP are listed in the Subsections 4.2.1 and 4.2.2, respectively.

4.2.1 Related SPP Schemes for Hiding Data

As an SPP scheme, Hong (2012) [31] presented a median edge detection based predictor where the predictor first detects a horizontal, vertical or diagonal edge that passes through it by comparing the values of the immediate left, top and left-top, i.e., corner, pixels. If the edge is vertical and passes through the left of the predicting pixel, the predictor picks the value of the top pixel as the prediction value of the working pixel. For the existence of a horizontal edge above the predicting pixel, it predicts the working pixel by the value of the left pixel. The expression of (left pixel value + top pixel value — corner pixel value) is computed for

generating the predicted value of the current pixel when no edge is detected. Ou *et al.* (2013) [72] proposed a methodology that predicts a pixel by repeatedly applying a partial differential equation (PDE) on four of its neighbor pixel values located at the top, right, bottom, and left. The scheme proposed by Yang *et al.* (2013) [107] predicted each pixel by taking the weighted average of those four neighbor pixels. Ma *et al.* (2015) [63] presented a scheme where multiple predictors were applied to predict each pixel value. Among these predicted values, the more accurate one is chosen as an optimal prediction value for the working pixel. Nevertheless, this predicted value does not ensure that this will generate an embeddable error. Chen *et al.* (2013) [11] employed two asymmetric predictors separately. First, a predictor is applied and the prediction errors are measured. The embedding process implants bits into the highest appeared error by modifying this and its greater valued errors in the right direction in the error histogram. Next, the second predictor is applied and the embedding process is executed. In this embedding process, the highest appeared and its smaller valued errors are modified in the left direction in the error histogram. The main objective of the scheme is to reduce the amount of total error shifting. Indeed, this is a double-time data embedment process where, as if, two machines are individually embedding in the same image at two separate times. The scheme proposed by Wang *et al.* (2013) [97] applies two separate predictors where one predictor predicts a group of pixels in each image block, say, boundary pixels and another predicts the rest of them. The prediction values of these two disjoint predictors are not dependent on one another for any operational purpose. Details of these prediction schemes and their limitations are given in Chapter 2.

4.2.2 Related BPP Schemes for Data Hiding

As a BPP scheme, Tsai *et al*. (2009) [88] estimated the values of all pixels within a block from the value of its center pixel. Hong and Chen (2010) [32] improved this prediction scheme by associating the center pixels of four neighbor blocks along with its own center pixel in their prediction rules. These two schemes leave the block centers, known as the basic pixels, unchanged for the convenience of reapplying the same estimation process at the receiver. Lu and Huang (2014) [59] proposed a scheme for increasing the embedding capacity by bringing the basic pixels of all blocks in an imaginary plane upon the image and applying the same prediction and data embedment process in these pixels. In Chapter 3, we prove that the schemes presented in [32, 59, 88] cannot predict well in texture images when the deep valley or the sharp hill exists in a frequent manner around the working block. To overcome this, we

propose a new prediction scheme based on Euclidean distances between each working pixel and every of its context basic pixels to improve the prediction accuracy and thus, to enhance the number of embeddable errors. The work is published in the proceedings of ANTS 2015. Another BPP based method proposed by Leung *et al.* (2013) [51] improved the prediction accuracy by predicting pixel values in a block with the median value of block pixels. Chang *et al.* (2015) [7] presented a scheme where data bits are implanted into the block truncation coded image by shifting the residual errors in the histogram. The operational methodologies for all of these schemes and the limitations have already been detailed in Chapter 2.

All the reviewed SPP based schemes [11, 31, 63, 72, 107] and BPP based schemes [7, 32, 40, 51, 59, 88] are experimented and compared with the scheme proposed in this chapter. The simulation results demonstrate that the embedding capacity in our proposed scheme has been improved notably compared with its competing schemes. Additionally, this scheme also improves the stego image quality.

4.3 Proposed Multi-Predictor Based Reversible Data Embedment Scheme

The proposed reversible data embedment (RDE) scheme employs n predictors, say A_k for $1 \leq k \leq n$, in its prediction phase for increasing the frequencies of two highest occurrence embeddable errors -1 and 0. The scheme is also called the multi-predictor scheme and shortly as the n-predictor scheme, e.g., 2-predictor, 3-predictor and 4-predictor schemes for the uses of two predictors, three predictors and four predictors, respectively. Let the n predicted values for a pixel $I_{i,j}$ at (i, j) location in the image in the n-predictor scheme are $P_{k,i,j}$. For example, if n=3, a cover pixel at (i, j) location is predicted by the predictors A_1, A_2 and A_3 separately and the predicted values are $P_{1,i,j}$, $P_{2,i,j}$ and $P_{3,i,j}$ respectively. The prediction error $E_{k,i,j}$ for a specific predictor A_k is measured by subtracting the predicted value from the processing cover pixel, $I_{i,j}$, i.e., $E_{k,i,j} = I_{i,j} - P_{k,i,j}$. Similarly, n prediction errors are measured against each cover pixel $I_{i,j}$. The proposed scheme first sequentially examines n prediction errors $E_{1,i,j}$, $E_{2,i,j,}$ ..., ..., $E_{n,i,j}$; if any of these errors hold a value r, where $r \in \{-1, 0\}$, then the first encountered r is considered as a final prediction error and this scheme stops the execution for this pixel and proceeds to the next pixel; otherwise the scheme generates m new errors, known as hybrid errors, by employing these n errors into m different linear relations. The scheme again looks for an error r inside these new m errors as well. If the error r is found, the first encountered -1 or 0 is taken

as an optimal prediction error. If the scheme is still unable to find an error of r, the minimum of the m hybrid errors is regarded as the optimal prediction error. The optimal error generation process is repeated for all the pixels. Let the generated optimal errors for the image pixels are $e_{i,j}$. The proposed scheme increases the frequencies of the two errors of −1 and 0 in the computed prediction errors $e_{i,j}$.

Algorithm 4.1: CoverOptimalErrorGeneralized

Prediction errors list: It contains n prediction errors.

Hybrid errors list: That contains m hybrid errors.

Combine errors list: It contains n prediction errors and then m hybrid errors, thus total n+m errors.

Step 1: *It find an error from n prediction errors in the prediction error list that valued to -1 or 0, if such error exists. If multiple errors arises with the value of -1 or 0, the first one is regarded as an optimal error for the working pixel. Say the position of the optimal error in the prediction errors list is k, the position of the optimal error in the combine error list is also k. It saves the value of k to $AP_{i,j}$.*

Step 2: *If none of the n prediction errors of the prediction error list contains -1 or 0, it moves to the hybrid errors list. It finds the first encountered error from m hybrid errors, if such a hybrid error exists, which has a value of -1 or 0. If such an error is found in the hybrid errors list, it is regarded as an optimal error for the working pixel. Say, the position of that optimal error in the hybrid errors list is k. Then, the position of optimal error in the combine errors list is n+k. It saves the value of n+k to $AP_{i,j}$.*

Step 3: *If both the steps fail to generate an optimal error, it measures the minimum of the hybrid errors as an optimal error value. If multiple values become the minimum, the first one is regarded as the optimal error. Let, the position of the optimal error in the hybrid errors list is k. The value of n+k, i.e., index value of the optimal error in the combine errors list, is saved to $AP_{i,j}$.*

Figure 4.1: A generalized algorithm for optimal error generation.

The optimal error generation module keeps a track of the acting predictor ($AP_{i,j}$), i.e., optimal error provided by the predictor for each pixel at (i, j) location, for the purpose of to be utilized by the data embedment module. The methodology of generating optimal prediction errors and AP list is presented as an algorithmic module in Figure 4.1. The encoder embeds

message bits into $e_{i,j}$ by its embedding rules as represented by the Eq. (4.2). After passing the data embedment phase, these modified optimal errors are termed as the stego errors, $\tilde{e}_{i,j}$. If the amount of modification made due to data embedment is $M_{i,j}$, then $\tilde{e}_{i,j} = e_{i,j} + M_{i,j}$. The encoder finally forms the Stego image $S_{i,j}$ by adding each of the stego errors, $\tilde{e}_{i,j}$ with the prediction value provided by the corresponding optimal error provided predictor. The parameter $AP_{i,j}$ guides the encoder to find this prediction value, as it is done by Eq. (4.3).

The embedding process works in a way so that each k^{th} predictor in the decoder end can predict the same value $P_{k,i,j}$ from the stego image without any ambiguity. Like the encoder, the decoder also generates n prediction errors and m hybrid errors. The process of measuring the optimal stego errors is outlined in the Section 4.3.1.2. The hidden message is extracted from these optimal stego errors by data extraction rules represented in Eq. (4.5). Thereafter, the original prediction errors are reconstructed to retrieve the cover pixels.

The Sections 4.3.1 and 4.3.2 narrate the whole steganographic process. For a better understanding of the proposed scheme, first, a 2-predictor based optimal error computation method and the data embedment process are explained in Section 4.3.1. Finally, the generalized optimal error computation method is presented for n predictors in Section 4.3.2. The symbols M, e, $\tilde{e}$, E_k, I and S are used as the matrices of $M_{i,j}$, $e_{i,j}$, $\tilde{e}_{i,j}$, $E_{k,i,j}$, $I_{i,j}$ and $S_{i,j}$, respectively, in the explanation of the proposed scheme and error computation method.

4.3.1 Two Predictors Based RDE Scheme

The conventional single predictor based RDE scheme, as shown in Figure 4.2(a), utilizes only one predictor during their data embedment process. On the contrary, the proposed two-predictor based RDE scheme applies two different predictors in its prediction phase, as shown in Figure 4.2(b). The two-predictor based scheme predicts each cover pixel separately by using each of the two predictors A_k, where $k \in \{1, 2\}$ to obtain two prediction values. Let the predicted values and the prediction errors of these two predictors A_1 and A_2 are P_1, P_2, and E_1, E_2, respectively.

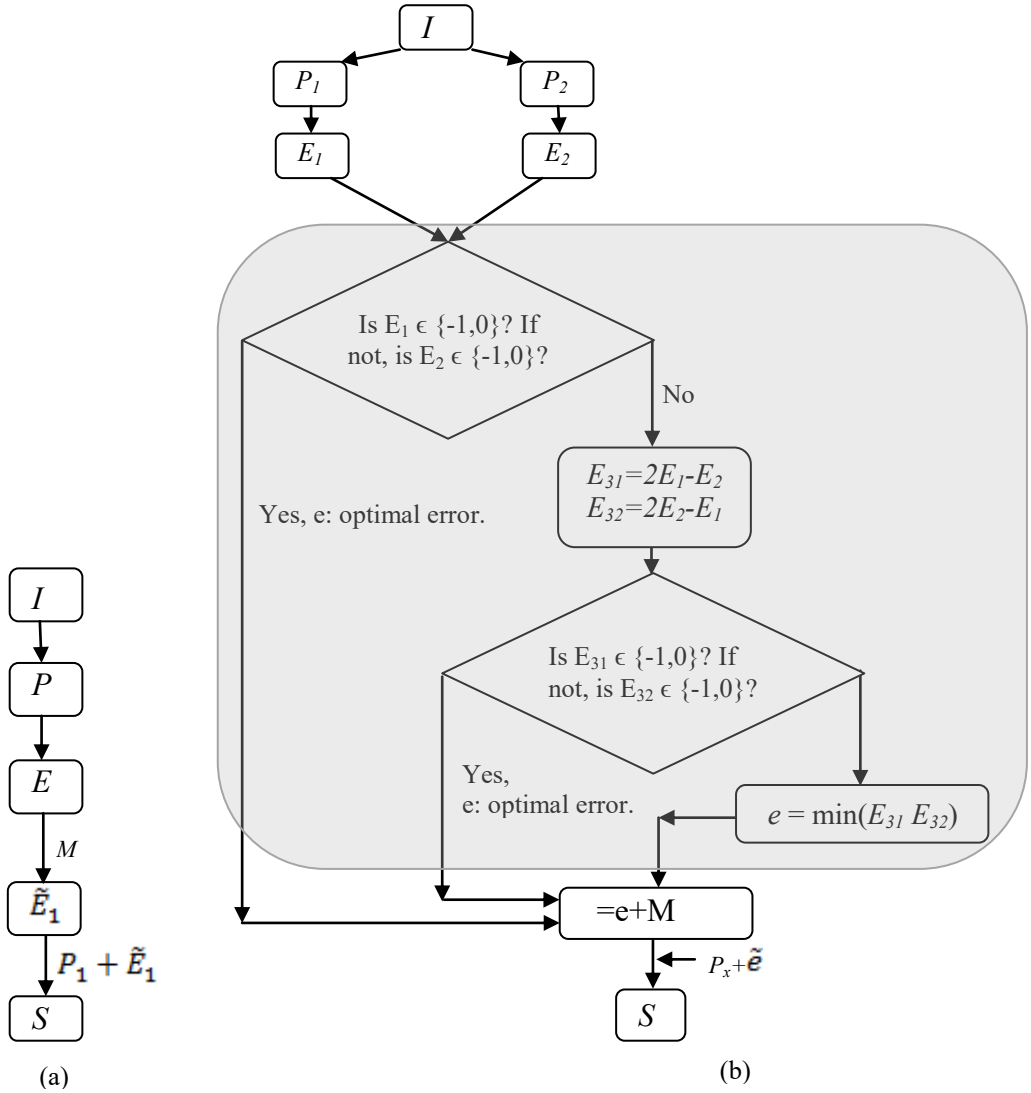


Figure 4.2: Single predictor vs. two-predictor based RDE Scheme: (a) Single predictor based RDE scheme; and (b) Two-predictor based RDE scheme.

4.3.1.1 Generating Hybrid Errors

The two error matrices E_1 and E_2 are used to generate new errors, E_{3x}, termed as hybrid errors, from an expression $E_{3x} = \beta_{1,x}B_{1,x}E_1 + \beta_{2,x}B_{2,x}E_2$, where $x \in \{1, 2\}$; $\beta_{1,x}B_{1,x} + \beta_{2,x}B_{2,x} = 1$; $\beta_{1,x}, \beta_{2,x} \in \{-1, 1\}$; and $B_{1,x}$, $B_{2,x}$ are of integer-valued. The constraint of $\beta_{1,x}B_{1,x} + \beta_{2,x}B_{2,x} = 1$ ensures that after choosing the value for $\beta_{1,x}$, $\beta_{2,x}$, $B_{1,x}$ and $B_{2,x}$, the hybrid error E_{3x} will be E if we consider all errors like E_1, E_2 as E. This means that the hybrid error E_{3x} is just another error. Although, $B_{1,x}$ and $B_{2,x}$ can hold any value maintaining the constraint of

$\beta_{1,x}B_{1,x} + \beta_{2,x}B_{2,x} = 1$, for the processing simplicity $B_{1,x}$ and $B_{2,x}$ are bounded here to hold only one of the values, say, {1, 2}. Eqs (4.1.1) and (4.1.2) are executed to produce two hybrid errors E_{31} and E_{32} (in the case of $B_{1,x}$, $B_{2,x} \in \{1, 2\}$).

$$\left.\begin{aligned} E_{31} &= 2E_1 - E_2 \qquad (4.1.1) \\ E_{32} &= 2E_2 - E_1 \qquad (4.1.2) \end{aligned}\right\} \tag{4.1}$$

Algorithm 4.2: CoverOptimalErrors2Pred(E_1, E_2, E_{31}, E_{32})

1. *If* $E_1 = 0$ *or* $E_1 = -1$ *then*
2. $E_3 = E_1$ *and* $AP_{i,j} = 1$
3. *Else if* $E_2 = 0$ or $E_2 = -1$ *then*
4. $E_3 = E_2$ *and* $AP_{i,j} = 2$
5. *Else if* $E_{31} = 0$ *or* $E_{31} = -1$ *then*
6. $E_3 = E_{31}$ *and* $AP_{i,j} = 3$
7. *Else if* $E_{32} = 0$ *or* $E_{32} = -1$ *then*
8. $E_3 = E_{32}$ *and* $AP_{i,j} = 4$
9. *Else*
10. $E_3 = \min(E_{31}, E_{32})$
11. If $E_{31} \le E_{32}$ *then*
12. $AP_{i,j} = 3$
13. *Else*
14. $AP_{i,j} = 4$
15. *End if*
16. *End if*

Figure 4.3: Generation of optimal prediction errors.

In the Eq.(4.1.1) $B_{1,x}$=2, $B_{2,x}$=1, $\beta_{1,x}$=1 and $\beta_{2,x}$=-1, whereas in the Eq. (4.1.2) $B_{1,x}$=1, $B_{2,x}$=2, $\beta_{1,x}$=-1 and $\beta_{2,x}$=1.

4.3.1.2 Generating Optimal Prediction Errors

Using Algorithm 4.2 of Figure 4.3, each optimal prediction error E_3 (temporarily stored in this variable before saving to *e*) is computed from the two prediction errors E_1, E_2 and the two hybrid errors E_{31}, E_{32}. A parameter *AP* keeps track of one of the E_1, E_2, E_{31} and E_{32} for each of the pixels that provides the optimal error. For this purpose, first, the prediction errors are

assigned a numerical ID which ranges from 1 to *n*, here *n*=2 as it is for two predictors scheme. For example, ID is 1 for E_1 and 2 for E_2. Then, next m numerical values range from *n*+1 to *n*+*m* are assigned to *m* hybrid errors. For *m* = 2, the hybrid errors of E_{31} and E_{32} are given a numerical value of 3 and 4, respectively. The ID of the optimal error presenting one is assigned as the value of the *AP*, e.g., $AP_{i,j}$ =1 if $E_{1,i,j}$ is equal to -1 or 0, $AP_{i,j}$ =2 if $E_{2,i,j} \in \{-1, 0\}$,and so on, where $E_{x,i,j}$ is the prediction error generated by the predictor A_x for the pixel at (*i*, *j*). An illustration of the stated optimal prediction error generation process is provided in the shaded area of Figure 4.2(b) as well as in Example 4.1. The Figure 4.2(b) is, indeed, a snapshot of the proposed embedding process that flows based on the Eq. (4.1). The former rhombus in the shaded area in the figure states that the scheme first sequentially checks whether E_1 or E_2 is equivalent to one of the values of embeddable errors of -1 and 0. If a value is found, the first matched value is regarded as the optimal prediction error value against the processing pixel. Otherwise, the Eq. (4.1) is applied to generate two hybrid errors E_{31} and E_{32}, as shown in the first rounded rectangle in the shaded area of the figure. Like E_1 and E_2, an optimal error is computed from E_{31} and E_{32} if such a one is found, as shown in the second rhombus of the shaded area. If the hybrid errors still cannot provide the optimal error, then it is computed from the minimum of the two hybrid errors in the last rounded rectangle of the shaded area. Finally, message bits are embedded into these optimal errors.

4.3.1.3 Message Embedment and Stego Image Generation

The computed optimal errors E_3 are assigned to a two dimensional matrix $e_{i,j}$, i.e. $e_{i,j} = E_{3,i,j}$ each for one pixel of the image. Using the Eq. (4.2), all the message bits are then embedded into $e_{i,j}$ in the encoder side. After the concealment of information, the modified errors $\tilde{e}_{i,j}$ are assigned to $\tilde{E}_{3,i,j}$, i.e., $\tilde{E}_{3,i,j} = \tilde{e}_{i,j}$. As a final step in the encoder end, the Eq. (4.3) is used to generate the stego image *S*. The pixels associated with the prediction rules remain as unchanged. The process of the data embedment and the stego image generation is explained in Example 4.2.

$$\tilde{e}_{i,j} = \begin{cases} e_{i,j}\omega s & \text{if } e_{i,j} = 0 \text{ or } e_{i,j} = -1 \\ e_{i,j}\omega 1 & \text{if } e_{i,j} > 0 \text{ or } e_{i,j} < -1 \end{cases} \tag{4.2}$$

where, $\omega = sign\_of(\tilde{e}_{i,j})$.

$$S_{i,j} = \begin{cases} P_{1,i,j} + \tilde{E}_{3,i,j} & \text{if } AP_{i,j} = 1 \quad (4.3.1) \\ P_{2,i,j} + \tilde{E}_{3,i,j} & \text{if } AP_{i,j} = 2 \quad (4.3.2) \\ P_{1,i,j} + \tilde{E}_{3,i,j} - E_{1,i,j} + E_{2,i,j} & \text{if } AP_{i,j} = 3 \quad (4.3.3) \\ P_{2,i,j} + \tilde{E}_{3,i,j} + E_{1,i,j} - E_{2,i,j} & \text{if } AP_{i,j} = 4 \quad (4.3.4) \end{cases} \tag{4.3}$$

Example 4.1: Generating optimal prediction errors.

Assume an image block as it is shown in Figure 4.4(*a*). *The center pixel known as the basic pixel of the block is 55. Let the value of the basic pixels of its four neighbor blocks, not depicted in this figure, situated on top, right, bottom and left are* 53, 54, 53 *and* 52, *respectively. The predictors of Tsai et al.'s*[88] *and Hong and Chen's* [32] *schemes are separately applied to the block pixels. Say, these two predictors are A_2 and A_2 correspondingly. The predicted values computed by A_1 and A_2 are tabulated in Figure* 4.4(*b*) *and* 4.4(*c*), *respectively. The respective prediction errors E_1 and E_2 are tabulated in Figure* 4.4(*d*) *and* 4.4(*e*), *correspondingly.*

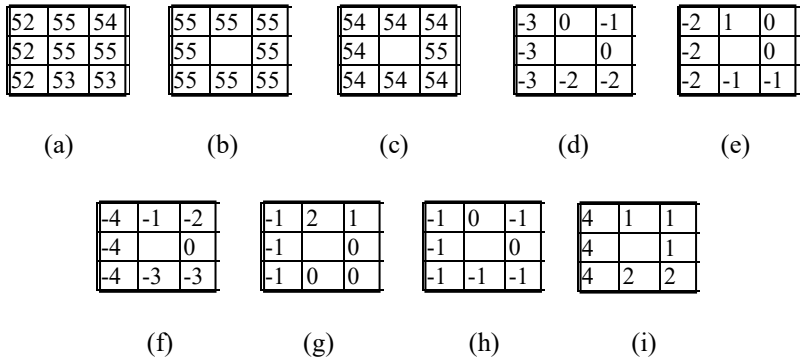


Figure 4.4: Generating optimal prediction errors in the two-predictor scheme: (a) an instance of a cover image block; (b) the predicted values generated by Tsai *et al*. (2009)[88]; (c) the predicted values generated by Hong and Chen (2010)[32]; (d) the prediction error values generated by Tsai *et al*. (2009)[88]; (e) the prediction error values generated by Hong and Chen (2010)[32]; (f) hybrid error E_{31}; (g) hybrid errors E_{32}; (h) optimal errors E_3; and (i) list of applied predictors *AP*.

Hybrid prediction errors E_{31} and E_{32} are measured by applying Eqs (4.1.1) *and* (4.1.2) *and these are tabulated in Figure* 4.4 (*f*)-(*g*), *respectively. Finally, algorithm* 1 *in Figure* 4.1 *computes the optimal errors E_3. Figure* 4.4(*h*) *demonstrates the generated optimal errors. Figure* 4.4(*i*) *lists the values of AP. Each predictor generates* 8 *errors. Among these* 8 *errors, the number of embeddable errors, i.e.,* -1 *and* 0, *is* 38% (3/8) *in E_1,* 50% (4/8) *in E_2,* 25% (2/8)

in E_{31}, 75% (6/8) *in E_{32} and* 100% (8/8) *in E_3. This example shows that the two-predictor scheme has an increase of* 62% *and 50% in the generation of embeddable errors compared to the same for A_1 and A_2 predictor based scheme.*

Example 4.2. The data embedment and stego image generation processes.

The E_3 computed in Example 4.1 *is copied into $e_{i,j}$ in Figure* 4.5(*a*)*. Let a chunk of the message stream that is to be embedded is* 11101001*. The Eq. (*4.2) *is used to implant these message bits into $e_{i,j}$. The Figure* 4.5(*b*) *demonstrates stego errors $\tilde{e}_{i,j}$ generated. The data embedment process is started from the upper left corner and the scheme embeds bits into the errors row-by-row. For the convenient of stego image generation, some information such as the AP list, P_1, P_2, E_1 and E_2 are copied here in Figure* 4.5(*c*)-(*g*) *from the Example* 4.1*. The portions of Eq. (*4.3) *such as* 4.3.1, 4.3.2, 4.3.3 and 4.3.4 *that are used for stego image generation are noted in Figure* 4.5(*h*)*. Figure 4.5(i) depicts the Stego block generated after applying the embedding process.*

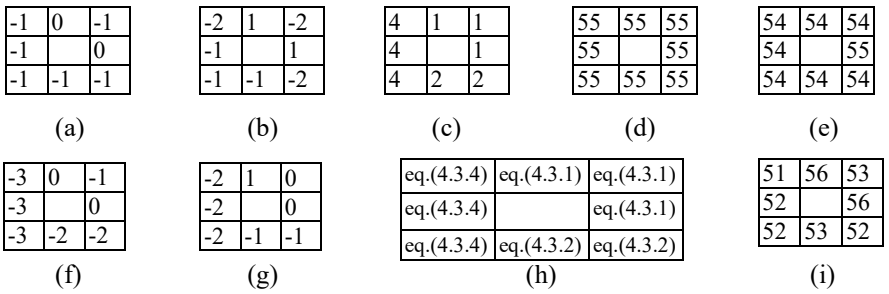


Figure 4.5: The process of data embedment into the optimal errors and the stego image generation: (a) optimal errors; (b) stego errors $_{i,j}$,; (c) acting predictors $AP_{i,j}$; (d) predicted values using A_1 predictor; (e) predicted values using A_2 predictor; (f) prediction errors generated for A_1 predictor; (g) prediction errors generated for A_2 predictor; (h) list of equations applied during Stego pixel generation; and (i) Stego values.

4.3.1.4 Message Extraction and Cover Image Reconstruction

The decoder applies the same *n* predictors in the stego image to predict each stego pixel separately. Let these predicted values be P_k^d . The encoder did not change the pixels, which

took part in the prediction rules. This implies that these P_k^d and P_k are the same, i.e. $P_k = P_k^d$ for all k. The scheme measures the stego prediction errors $\tilde{E}_{k,i,j}$ by $\tilde{E}_{k,i,j} = S_{i,j} - P_{k,i,j}^d$, e.g. $\tilde{E}_{1,i,j} = S_{i,j} - P_{1,i,j}^d$ and $\tilde{E}_{2,i,j} = S_{i,j} - P_{2,i,j}^d$. The stego hybrid errors $\tilde{E}_{31}$ and $\tilde{E}_{32}$ are generated by using Eqs. (4.4.1) and (4.4.2), respectively.

$$\left.\begin{aligned} \tilde{E}_{31} &= 2\tilde{E}_1 - \tilde{E}_2 \qquad (4.4.1) \\ \tilde{E}_{32} &= 2\tilde{E}_2 - \tilde{E}_1 \qquad (4.4.2) \end{aligned}\right\} \tag{4.4}$$

The AP and the optimal stego errors $\tilde{E}_3$ are generated by the decoder. This optimal error generation process is depicted in the flowchart in Figure 4.7. This figure states the process of generating the AP and the optimal stego errors for n predictors rather than two. Nevertheless, this is fully realizable for n=2. In this process, at first, the decoder generates the stego errors $\tilde{E}_1$ to $\tilde{E}_n$ and hybrid errors $\tilde{E}_{n1}$ to $\tilde{E}_{nm}$, here n=2 and m=2, against each stego pixel. The encoder then arranges these stego and hybrid errors in two separate lists. An ID is assigned to each error as the encoder did it. The decoder sequentially searches in the stego errors, $\tilde{E}_1$ to $\tilde{E}_n$ to find an encountered error of the value of -1 or 0. An error of -1 or 0 can be only found if this stego error conceived a bit 0 at the encoder end. If the decoder finds such an error, this is regarded as an optimal stego error and the corresponding ID of the error is stored to $AP_{i,j}$. If the decoder does not find an error value of -1 or 0, it looks for a stego error with the value of -2 or 1. This is the case, when a prediction error of -1 or 0 was modified by implanting a message bit of 1 by the encoder. If the check finds a stego error with the value of -2 or 1, the decoder saves it as an optimal stego error to $\tilde{E}_{3,i,j}$ and the corresponding ID to $AP_{i,j}$. If the decoder fails again, this implies that none of the stego prediction errors, i.e., none of $\{\tilde{E}_1, \tilde{E}_2, ..., \tilde{E}_n\}$, conceives any message bit. In that circumstance, the scheme moves its execution pointer to hybrid errors to find an optimal stego error from them. The scheme repeats the whole process with the stego hybrid errors as it is done for stego prediction errors, i.e., it first checks in the hybrid errors for an error value of -1 or 0; if it fails, the scheme checks for an error value of -2 and 1. When a matching case is found, the scheme saves the error as an optimal stego error to $\tilde{E}_{3,i,j}$ and its ID to $AP_{i,j}$. If the scheme is still unable to find an error of -2, -1, 0, and 1 from these checks, this indicates that the pixel did not conceive any message bit. The decoder then collects the minimum of the hybrid errors as an optimal stego error and store the corresponding

ID to $AP_{i,j}$. Thus, the decoder calculates the optimal stego errors, $\tilde{E}_3$, and the full AP for the image pixels.

Assigning these optimal stego errors $\tilde{E}_3$ into $\tilde{e}_{i,j}$, Eqs. (4.5) and (4.6) are executed to extract each message bit s and the cover error $e_{i,j,}$ respectively. Finally, Eq. (4.7) constructs the cover image I. An illustration of message extraction and cover image reconstruction is provided in Example 4.3.

$$s = \begin{cases} 0 & \text{if } \tilde{e}_{i,j} = 0 \text{ or } \tilde{e}_{i,j} = -1 \\ 1 & \text{if } \tilde{e}_{i,j} = 1 \text{ or } \tilde{e}_{i,j} = -2 \end{cases} \tag{4.5}$$

$$e_{i,j} = \begin{cases} \tilde{e}_{i,j} \odot 1 & \text{if } \tilde{e}_{i,j} > H_p \text{ or } \tilde{e}_{i,j} < H_n \\ \tilde{e}_{i,j} & \text{Otherwise} \end{cases} \tag{4.6}$$

$$I_{i,j} = \begin{cases} P_1 + e_{i,j} & \text{if } AP_{i,j} = 1 \quad (4.7.1) \\ P_2 + e_{i,j} & \text{if } AP_{i,j} = 2 \quad (4.7.2) \\ P_1 + e_{i,j} - \tilde{E}_1 + \tilde{E}_2 & \text{if } AP_{i,j} = 3 \quad (4.7.3) \\ P_2 + e_{i,j} + \tilde{E}_1 - \tilde{E}_2 & \text{if } AP_{i,j} = 4 \quad (4.7.4) \end{cases} \tag{4.7}$$

where $\odot = sign\_of(-1 \times \tilde{e}_{i,j})$, $H_p = 0$ and $H_n = -1$

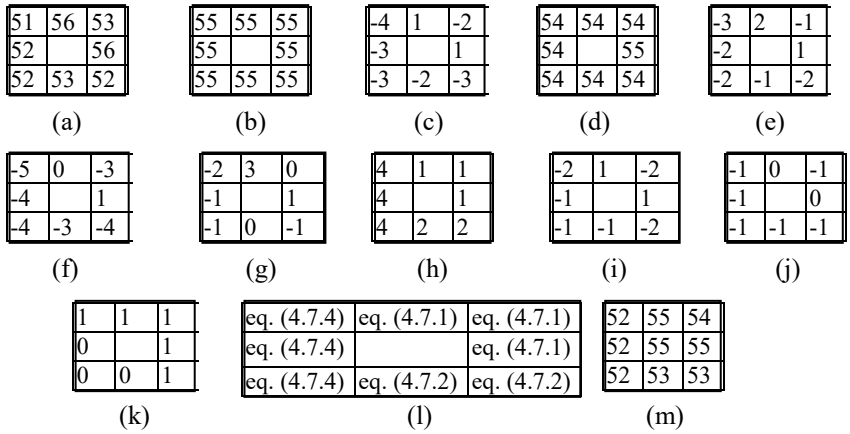


Figure 4.6: Optimal error generation, message extraction and cover reconstruction process: (a) the stego image block S; (b) predicted values P_1 using A_1 predictor; (c) prediction errors $_1$ from A_1; (d); predicted values P_2 using predictor A_2; (e) prediction errors $_2$ from A_2; (f) hybrid errors of $_{31}$, produced by Eq.(4.4.1); (g) hybrid errors of $_{32}$, computed by Eq. (4.4.2); (h) AP list, formed by Figure 4.1; (i) optimal stego errors $_3$; (j) recontracted cover errors $e_{i,j}$ by using Eq. (4.6); (k) extracted message bits M by using Eq. (4.5); (l) applied equations in each cell; and (m) the cover pixels.

Example 4.3. Message extraction and cover image reconstruction process from the stego image

The stego block is reprinted in Figure 4.6*(a) from the Example* 4.2*. The same predictors A_1 and A_2 are again applied in the stego block. Figure* 4.6*(b) and* 4.6*(d) list the predicted values P_1 and P_2, respectively. The corresponding stego prediction errors $\tilde{E}_1$ and $\tilde{E}_2$ are shown in Figure* 4.6*(c) and* 4.6*(e), respectively. Eqs.* (4.4.1) *and* (4.4.2) *generate the stego hybrid errors $\tilde{E}_{31}$ and $\tilde{E}_{32}$, which are depicted in Figure* 4.6*(f) and* 4.6*(g), respectively. The optimal stego errors $\tilde{E}_3$* and the list of *AP are computed by using the process stated in Figure* 4.7*. The AP list and all the optimal stego errors are demonstrated in Figure* 4.6*(h) and* 4.6*(i), respectively. Eq.* (4.5) *is used to extract the message of 11101001 from $\tilde{E}_3$ of Figure* 4.6*(i). The extracted message stream is shown in Figure* 4.6*(k). The original cover errors are computed by using Eq.* (4.6)*. These cover errors are tabulated in Figure* 4.6*(j). The components of the Eq.* (4.7)*, i.e. Eqs.* (4.7.1) *to* (4.7.4) *are applied in each cell during the stego image generation, are shown in Figure* 4.6*(l). The components of Eq.* (4.7) *reform the cover image block, as it is shown in Figure* 4.6*(m).*

4.3.2 Multi-Predictor Based RDE Scheme

The *n*-predictor scheme is generalized to work with *n* number of predictors, where *n* is an arbitrary positive number. The scheme first predicts each pixel value by each of the *n* predictors and then measures *n* prediction errors for each pixel u. Say, these predicted values and the prediction errors are $P_1, P_2, ..., P_n$ and $E_1, E_2, ..., E_n$, respectively. The rules of breeding the hybrid prediction errors are generalized by the following Eq. (4.8).

$$E_{n+1,m} = \beta_{1,m} B_{1,m} E_{1,m} + \ldots\ldots + \beta_{n,m} B_{n,m} E_{n,m}, \tag{4.8}$$

where $\beta_{1,m} B_{1,m} + \beta_{2,m} B_{2,m} + \ldots\ldots + \beta_{n,m} B_{n,m} = 1$, each of $\{\beta_{1,m}, \beta_{2,m}, \ldots\ldots, \beta_{n,m}\}$ can hold a value from {1, -1} and *m* is the number of possible expressions represented by the Eq. (4.8). Primarily, 1 is assigned to each value of $B_{k,m}$, for k = 1 to *n*. If *n* is an even number, then in each *m* expression only one $B_{k,m}$ will be updated by the value of 2 for the purpose of holding

the relation of $\beta_{1,m}B_{1,m}+\beta_{2,m}B_{2,m}+\ldots\ldots+\beta_{n,m}B_{n,m}=1$. It is observed that $B_{1,1}=2$, $\beta_{1,1}=1$, $B_{2,1}=1$ and $\beta_{2,1}=-1$ are used to generate Eq. (4.4.1); while $B_{1,2}=1$, $\beta_{1,2}=-1$, $B_{2,2}=2$ and $\beta_{2,2}=1$ are used to generate Eq. (4.4.2). A list of possible $E_{n+1,m}$ for 3, 4 and 5 predictors, i.e. n=3, n=4 and n=5, is shown in Table 4.1.

A modified version of the Algorithm 1 of Figure 4.1 is employed to compute an optimal error and the AP value from $E_1, E_2, \ldots\ldots, E_n$ and $E_{n+1,m}$ errors. The Algorithm 1 is designed to work with two predictors and the algorithm checks the values of E_1, E_2, E_{31} and E_{32} sequentially. In the generalized version, the algorithm serially checks n prediction errors, i.e. E_1, E_2, ... , E_n, and then m hybrid errors, i.e. $E_{n+1,1}$, $E_{n+1,2}$, ..., $E_{n+1,m}$ to find the optimal error. The corresponding ID of the error is recorded in AP. After computing all the optimal errors and their respective AP, the scheme assigns E_{n+1} to $e_{i,j}$. The embedding is done into the errors of $e_{i,j}$ by using the Eq. (4.2). The stego errors $\tilde{e}_{i,j}$ are then constructed by using the Eq. (4.2), are copied to $\tilde{E}_{n+1,m}$. Finally, the Eq. (4.9) produces the stego image.

$$S_{i,j}=\begin{cases} P_{A_{i,j}}+\tilde{E}_{n+1} & \text{if } AP_{i,j}\le n \\ I_{i,j}+\tilde{E}_{n+1}-E_{n+1} & \text{Otherwise} \end{cases} \tag{4.9}$$

In the decoder end, the data extractor generates n stego prediction errors $\tilde{E}_1$, ..., $\tilde{E}_n$ and m stego hybrid errors $\tilde{E}_{n+1}$, ..., $\tilde{E}_{n+m}$ for each of the processing stego pixels. Like the 2-predictor method, the decoder computes AP list and the optimal stego errors $\tilde{E}_{n+1}$ for all the image pixels. The Eq. (4.5) extracts the every bit m of M from $\tilde{E}_{n+1}$. The Eq. (4.6) reconstructs the cover errors $e_{i,j}$ from these optimal stego errors. Finally, Eq. (4.10) constructs the cover image in the n-predictor scheme.

$$I_{i,j}=\begin{cases} P_{A_{i,j}}+e_{i,j} & \text{if } AP_{i,j}\le n \\ S_{i,j}+e_{i,j}-\tilde{E}_{n+1} & \text{Otherwise} \end{cases} \tag{4.10}$$

The optimal stego error generation process is depicted in Figure 4.7. The functionality of the flowchart is explained in the Section 4.3.1.4. For clarification of the proposed method, for an example, assume that the prediction errors of E_1, E_2 and E_3 generated by three separate predictors are -2, -1 and 0, respectively. The hybrid errors of E_{41}, E_{42} and E_{43} are computed from the expressions stated in Table 4.1 are -1, -3 and 1, respectively and the IDs of the errors

of E_1, E_2, E_3, E_{41}, E_{42} and E_{43} are 1, 2, 3, 4, 5 and 6, respectively. In the encoder end, the Algorithm 1 of Figure 4.1 generates an optimal error E_4 = -1 and AP=2. If '0' bit is embedded into this error, the stego error $\tilde{E}_4$ will be -1, i.e., not changed. The decoder is blind about the value of the optimal error $\tilde{E}_4$, the AP and the embedded bit. Nevertheless, the decoder is able to compute $\tilde{E}_1$=-2, $\tilde{E}_2$=-1 and $\tilde{E}_3$=0 and then $\tilde{E}_{41}$=-1, $\tilde{E}_{42}$=-3 and $\tilde{E}_{43}$=1. The IDs of the stego and hybrid errors, $\tilde{E}_1$, $\tilde{E}_2$, $\tilde{E}_3$, $\tilde{E}_{41}$, $\tilde{E}_{42}$ and $\tilde{E}_{43}$ are 1, 2, 3, 4, 5 and 6, respectively. In these errors, $\tilde{E}_2$ is the first encountered one that holds -1, i.e., $\tilde{E}_2 \in \{-1, 0\}$. Hence, the optimal stego error is $\tilde{E}_2$ which is -1 and the respective $AP_{i,j}$ is 2. The Eqs. (4.5) and (4.6) extract the message bit '0' from this error and reconstruct the error value to -1, respectively.

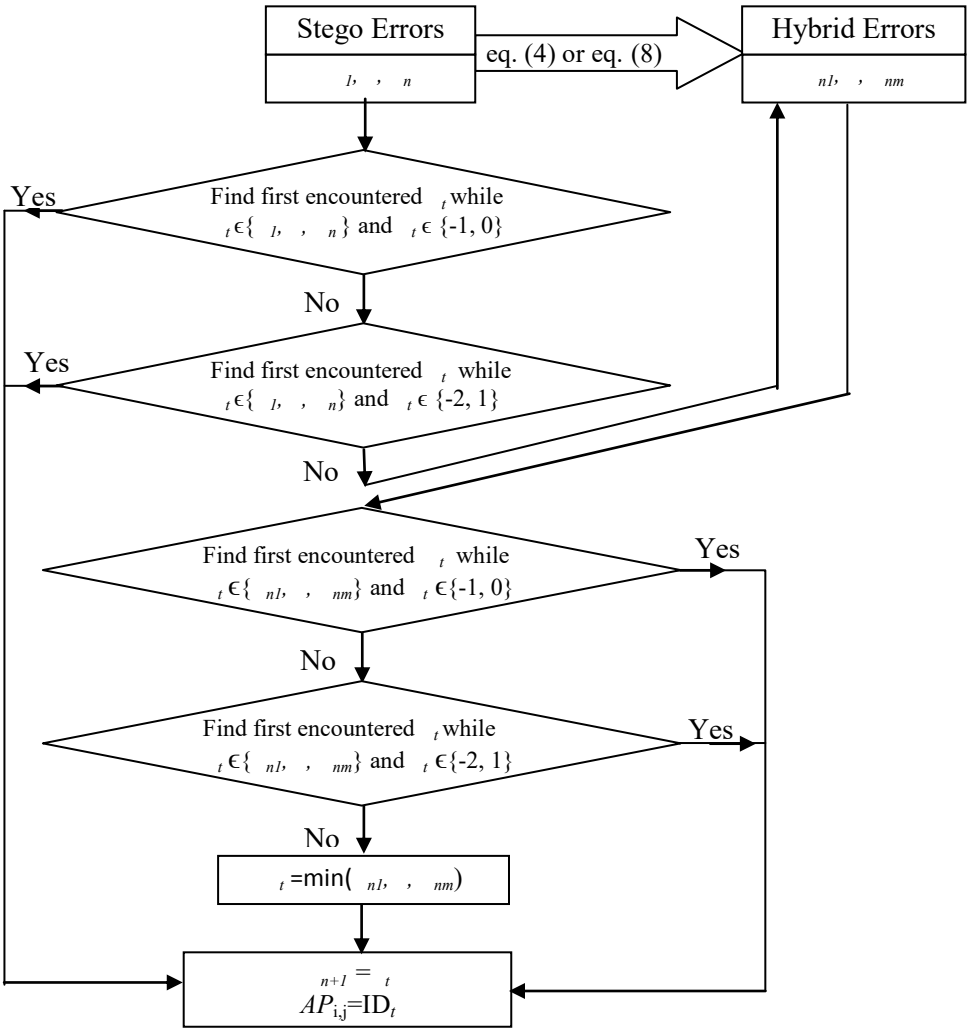


Figure 4.7: Generation of optimal prediction errors and AP list.

Table 4.1: Possible expression for hybrid errors $E_{n+1,m}$

m	Values of $E_{n+1,m}$ for		
	n=3	n=4	n=5
1	E_1-E_2+E_3	2E_1+ E_2- E_3-E_4	E_1+ E_2+E_3-E_4-E_5
2	E_1+E_2-E_3	E_1+ 2E_2-E_3- E_4	E_1+ E_2-E_3+E_4-E_5
3	-E_1+E_2+E_3	2E_1-E_2+E_3-E_4	E_1+E_2-E_3-E_4+E_5
4		E_1-E_2+ 2E_3- E_4	E_1-E_2+ E_3+E_4-E_5
5		2E_1- E_2-E_3+E_4	E_1- E_2+E_3-E_4+E_5
6		E_1- E_2-E_3+2E_4	E_1- E_2-E_3+E_4+E_5
7		-E_1- E_2+2E_3+ E_4	-E_1+ E_2+E_3+ E_4-E_5
8		-E_1- E_2+E_3+ 2E_4	-E_1+ E_2+E_3- E_4+E_5
9		-E_1+ 2E_2+E_3- E_4	-E_1- E_2+E_3+ E_4+E_5
10		-E_1+ E_2+2E_3- E_4	-E_1+ E_2-E_3+ E_4+E_5
11		-E_1+ 2E_2-E_3+E_4	
12		-E_1+ E_2-E_3+2E_4	

4.4 Result Analysis and Discussions

The main objective of the proposed prediction error based reversible data embedment scheme is to increase the embedding capacity as well as to enhance the quality of the stego image. To test these two performances-measuring parameters of the scheme, i.e., embedding capacity and stego image quality, experiments are conducted on 50 texture, 50 standard, 5000 CalTech, 50 natural and 50 satellite images in MATLAB. Both the SPP and the BPP based predictors are experimented on the above-listed image datasets. As the SPP based schemes, the predictors of Hong (2012) [31], Tai *et al.* (2009) [83], Yang *et al.* (2013) [107], Ma *et al.* (2015) [63] and Chen *et al.* (2013) [11] are tested and compared with the proposed *n*-predictor policies. The predictors of Tsai *et al.* (2009) [88], Hong and Chen (2010) [32], Lu and Huang (2014) [59], Kamal and Islam (2015) [40], Leung *et al.* (2013) [51] and Chang *et al.* (2015) [7] are employed in exploiting the BPP based schemes. Before applying the Chang *et al.*'s scheme, the image pixels are converted to truncation-coded values in our experiments. As a proposed *n*-predictor scheme, the 2-predictor, 3-predictor, 4-predictor and 5-predictor are analyzed. Generally, '*n*-predictor' is used in a sentence in this chapter to address it as a singular

collective noun; while the phrase '*n*-predictor policies' is used as the plural collective noun. In both the SPP and the BPP based prediction categories, the proposed *n*-predictor scheme dominates the other schemes by the values of all the performance measuring parameters like the payload, i.e., the embedding capacity, peak signal to noise ratio (PSNR) and structural similarity index matrix (SSIM). In the demonstration, the performance measuring values are plotted along the *y*-axis for the experimented images. Each legend in the figures represents a scheme. The meaning of the legends is tabulated in Table 4.2. These 20 schemes, as listed in Table 4.2, are separately experimented and the results are compared in this chapter.

4.4.1 Performance Analysis of the Predictors

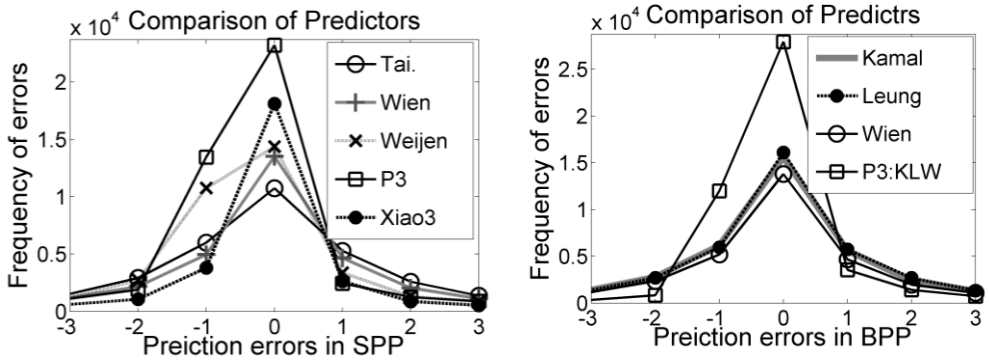
Taking the best of all, the *n*-predictor scheme enhances the quantity of the two embeddable errors, i.e., -1 and 0. Figure 4.8 depicts two prediction error histograms; one is for the SPP and other is for the BPP based predictors. The Figure 4.8(a) and Figure 4.8(b) demonstrate the prediction error histograms for several SPP and BPP based predictors, respectively. As a sample, only the 3-predictor is demonstrated in these figures. In both cases, i.e., in SPP and BPP, the 3-predictor scheme noticeably dominates the others by increasing the frequencies of the several highest appeared errors. The *n*-predictor scheme produces the highest frequency of the '0' valued error. The scenario is the same while comparing the frequencies of '−1' valued errors. Hence, the summation of these two embeddable errors is a lot more than the values of others. By improving the frequencies of the errors of −1 and 0, then-predictor scheme enhances the prediction accuracy as well. To test the prediction accuracy, prediction errors per predicting pixel (EPP) are measured where EPP is defined by Eq. (4.11). The values of the EPPs in the *n*-predictor policies are smaller than their competing schemes, as shown in Figure 4.9(a) and Figure 4.9(b).

$$EPP = \frac{\sum_{i=1}^{h}\sum_{j=1}^{w}\left|\left(I_{i,j} - P_{i,j}\right)\right|}{h \times w} \tag{4.11}$$

where $I_{i,j}$ is the predicting pixel, $P_{i,j}$ is the predicting value, h and w are the width and height of the image and |.| means the computation of absolute value.

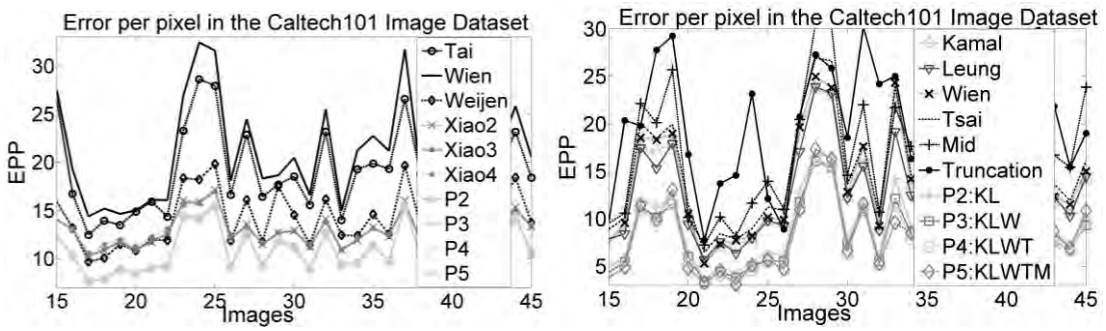
Table 4.2: An interpretation of the legends used in the Figure 4.8-4.11.

Sr.	Figure's Legend	For Scheme of
1	Tai	Tai *et al.* (2009)
2	Wien	Hong (2012)
3	Weijen	Yang *et al.* (2013)
4	Xiao2	Ma Xiaoxiao *et al.* (2015) with two predictors
5	Xiao3	Ma Xiaoxiao *et al.* (2015) with three predictors
6	Xiao4	Ma Xiaoxiao *et al.* (2015) with four predictors
7	Kamal	Kamal and Islam (2015)
8	Leung	Leung *et al.* (2013)
9	Wien	Hong and Chen (2010)
10	Tsai	Tsai *et al.* (2009)
11	Mid	Chen *et al.* (2013)
12	Truncation	Chang *et al.* (2015)
13	P2	Combined Wien and Weijen
14	P3	Combined Wien, Weijen and Tai
15	P4	Combined Wien, Weijen, Tai and Xiao2
16	P5	Combined Wien, Weijen, Tai, Xiao2 and Xiao3
17	P2:KL	Combined Kamal and Leung
18	P3:KLW	Combined Kamal, Leung and (Hong, 2010)
19	P4:KLWT	Combined Kamal, Leung, (Hong, 2010) and Tsai
20	P5:KLWTM	Combined Kamal, Leung, (Hong, 2010), Tsai and Mid



(a) (b)

Figure 4.8: Comparison of prediction errors: (a) SPP and (b) BPP techniques.



(a) (b)

Figure 4.9: A comparison of EPP among the schemes: (a) EPP in SPP based schemes; and (b) EPP in BPP based schemes.

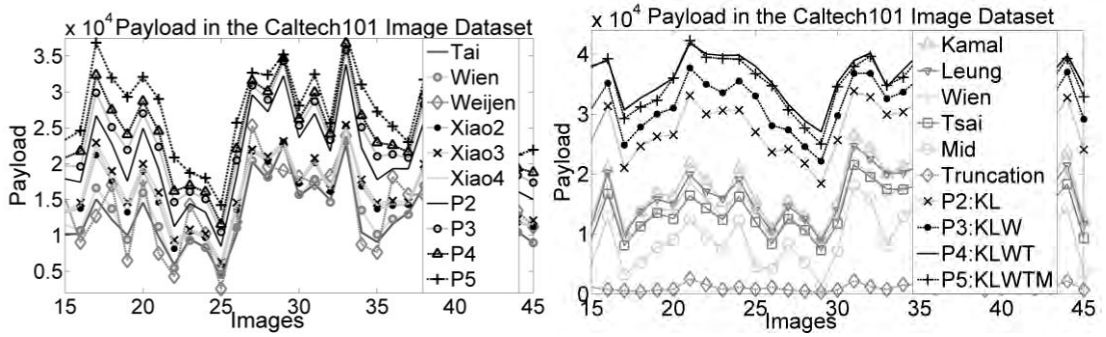
4.4.2 Analysis of the Embedding Payload

The total number of embedded bits, i.e., payload, by each scheme is measured for each image separately. Block or pixel skipping criteria used in different schemes (e.g., schemes of Hong (2012) [31]; Hong and Chen (2010)[32]; Kamal and Islam (2015)[40]; and Leung *et al.* (2013)[51]) is omitted in this chapter to allow the schemes to embed data bits at their maximum capabilities. The payloads, which are obtained in the images of the CalTech101 dataset, are delineated in Figure 4.10 along y-axis against the experimented images. The Figure 4.10(a) illustrates that the SPP based multi-predictor schemes like Ma *et al.*'s (e.g., Xiao2, Xiao3, Xiao4) and the proposed *n*-predictor policies (e.g., $P1$, $P2$, . . ., Pn) demonstrate superior payloads than all the single predictor based schemes, e.g. Tai, Wien, Weijen. The figure also states that between the multi-predictor schemes, the proposed *n*-predictor policies embed the highest amount of data and the values of embedded payloads are at a significant mark. It is also investigated that the amount of payloads raises for the use of more predictors, i.e., PL($P2$, I) > PL($P3$, I) > PL($P4$, I) > PL($P5$, I), where PL(X, I) represents the achieved payloads by the scheme of X in the image I. Similarly, Figure 4.10(b) exhibits the embedding payloads of *n*-predictor policies at a distinguishable higher level over the other BPP based schemes. These two figures prove that the *n*-predictor scheme outperforms in all the ways in providing better payloads.

To figure out the rate of improvement of *n*-predictor policy over the competing schemes, embedding gains are measured and analyzed as well. The embedding gains of the proposed *n*-predictor scheme over the compared methodologies in each image I for the implantation of the same message stream are measured in percentage by using the Eq. (4.12).

$$EmbeddingGain(I) = \frac{PL(X,I) - PL(Y,I)}{PL(Y,I)} 100\% \tag{4.12}$$

where $X \in \{$'*P2*', '*P3*', '*P4*', '*P5*', '*P2:KL*', '*P3:KLW*', '*P4:KLWT*', '*P5:KLWTM*'$\}$ and $Y \in \{$'*Tai*', '*Wien*', '*Weijen*', 'Xiao2', '*Xiao3*', '*Xiao4*', '*Kamal*', '*Leung*', '*Wien*', ' *Tsai*', '*Mid*', '*Truncation*'$\}$.



(a) (b)

Figure 4.10: Comparison of payloads among the schemes: (a) payloads in SPP based schemes; (b) payloads in BPP based schemes

Table 4.3: Embedding Gains of 4-predictor policy over the SPP based schemes in the CalTech image dataset.

Gain of 4-predictor over the ↓	**CalTech**	
	Min	**Max**
Tai	25	152
Wien	21	143
Weijen	19	519
Xiao2	21	71
Xiao3	18	56
Xiao4	19	83

As a sample, the minimum and the maximum gains of the 4-predictor based SPP and BPP policies are tabulated in Tables 4.3 and Table 4.4, respectively, only for the CalTech image dataset. All the minimum gains are positive valued. Several of the maximum gains are praiseworthy notable; even, these gains are some multiples of their competing schemes. The proposed scheme achieves the highest gains over the embedding into the truncated images by Chang *et al.* [7] for two reasons. Firstly, the data hider in [7] implants message bits into a truncated image, which is a compressed version of pixel value oriented. The compression

policy constructs the truncated image by encoding all pixels in each block into two integer values and preparing a binary map for the pixels. Consequently, the linear predictor, that is applied in [7], generates only two error residues for these two quantized integer values in a block and thus, the scheme embeds at most two bits per block in a truncated image. Secondly, the linear predictor in [7] operates on a group of pixels, while these pixels are collected from the separate blocks in the truncated image. These pixels, which come from separate blocks (i.e., from apart places), are more uncorrelated. The applied linear predictor then distributes its error residues over the error space more evenly rather than concentrating the majority of errors to a few values.

Table 4.4: Embedding Gains of 4-predictor policy over BPP based schemes in CalTech image dataset.

Gain of 4-predictor over the ↓	**CalTech**	
	Min	**Max**
Kamal	27	199
Leung	19	181
Wien	35	211
Tsai	27	226
Mid	54	1276
Truncation	1351	8349

4.4.3 Analysis of the Image Quality

The data embedment scheme alters the visual, the structural and the statistical information in the stego image regarding its cover image. The level of modification is estimated by using either a pixel difference measurement process or human visual based measurement processes. Two common mechanisms of pixel difference measurements are the mean-square-error (MSE) and PSNR. Two widely used human visual based measurement policies are the SSIM and the universal image quality index (UIQI). In this subsection, the image quality is measured and compared in terms of the PSNR and the SSIM values.

4.4.3.1 Analysis of PSNR Values

The PSNR of a stego image S of size $h \times w$, generated by embedding data bits into a cover image I of the same size, is measured. Thereafter, the PSNR loss (i.e., PSNR_LOSS is defined in Chapter 2) in each image due to data embedment is computed.

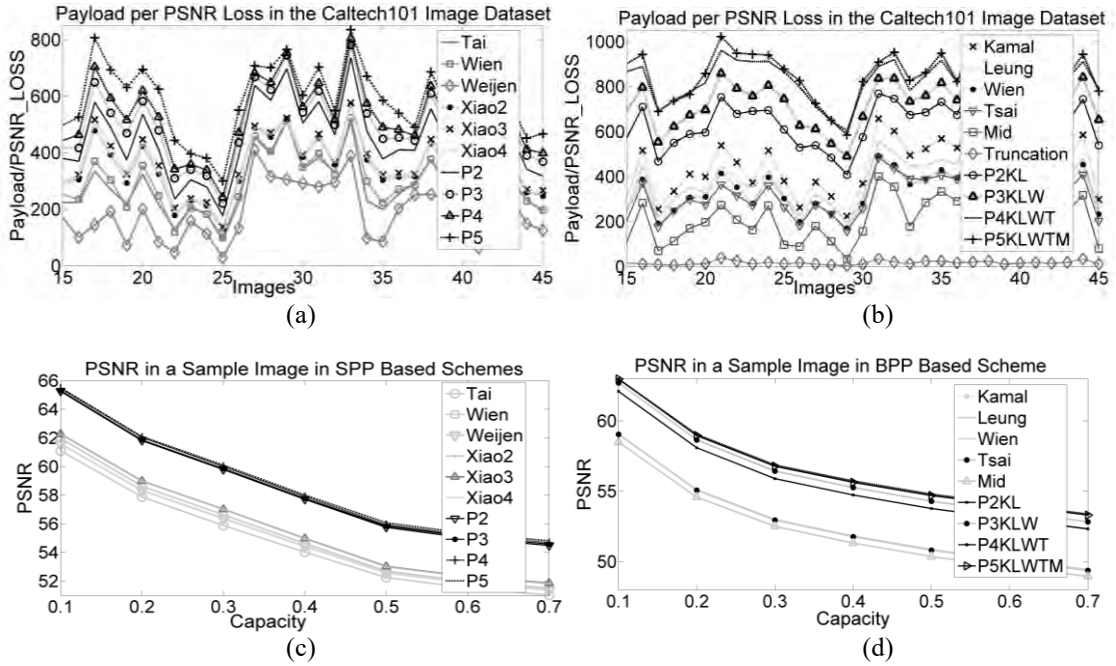


Figure 4.11: Comparison of PSNR values among the schemes: payload per PSNR loss in (a) SPP and (b) BPP based schemes; PSNR for different embedding capacities in (c) SPP and (d) BPP based schemes.

Figure 4.11(a) and Figure 4.11(b) depict the number of implanted bits per PSNR loss in the CalTech image dataset as a clipped portion of the results for the whole dataset. The values of payloads per PSNR losses are plotted along the y-axis for each image. The figures clearly depict that the *n*-predictor policies implant more bits for the same level of PSNR losses than all other competing schemes. This is also investigated that the amount of payload per PSNR_LOSS increases for each higher value of *n* in the *n*-predictor scheme. Figures 4.11(c) and Figure 4.11(d) demonstrate the PSNR values of different schemes along the y-axis for the same level of embedding capacity achieved in the CalTech image dataset. The PSNRs at each level of embedding capacity is measured by averaging the PSNR values of all the images. The figures state that the *n*-predictor policies provide higher PSNR values during the same amount of data embedment. Thus, Figure 4.11 establishes that the proposed scheme achieves better stego image quality regarding PSNR value.

4.4.3.2 Analysis of SSIM Values

SSIM represents the affair of structural similarity between two images by a single numerical value. To compare the cover and stego image quality, the values of SSIM and structural

dissimilarity index matrix (SDIM) between these two images are computed. The detail methodologies for computing the SSIM and the SDIM between a cover and a stego image are described in Chapter 2.

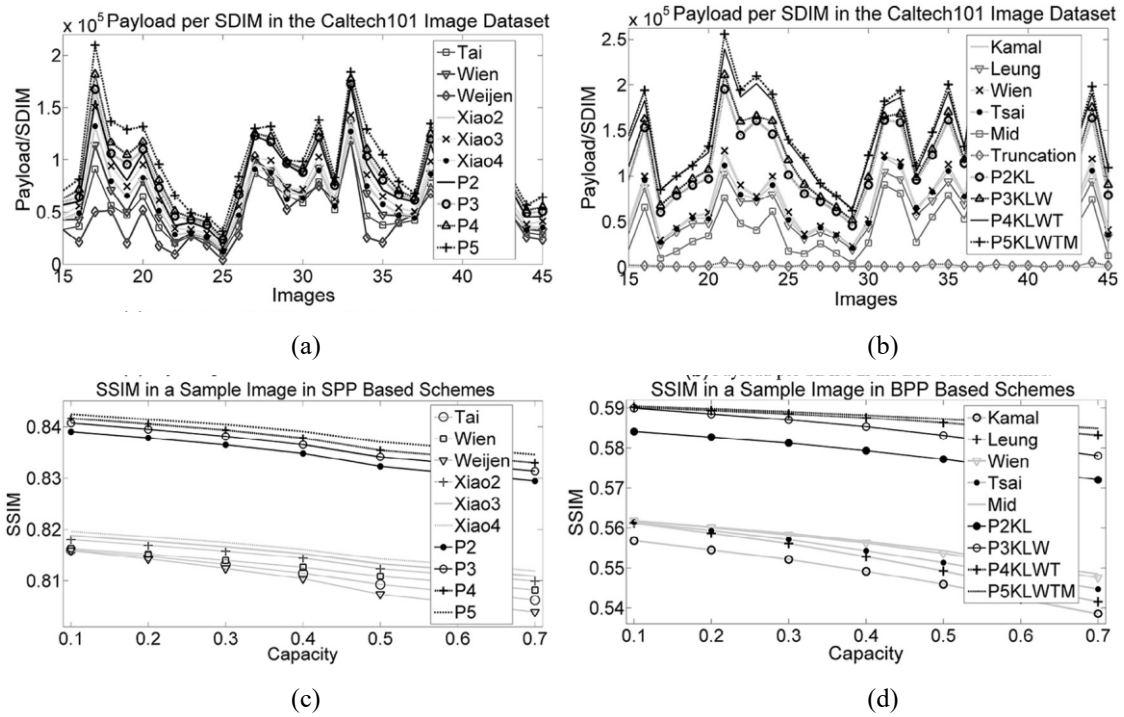


Figure 4.12: Comparison of embedding efficiency regarding the structural similarity index values among the schemes: payload per SDIM in different images in (a) SPP and (b) BPP based schemes; and SSIM at different capacities in (c) SPP and (d) BPP based schemes.

Figure 4.12(a) and Figure 4.12(b) depict the quantity of implanted bits per one unit loss of SDIM, in both the SPP and the BPP based schemes, respectively. The values of payload per SDIM are plotted along the y-axis for all the experimented images. The figures state that for the same amount of dissimilarities, all the *n*-predictor policies implant more quantity of bits and the amount of embedded bits increases for the increment in the number of applied predictors in the *n*-predictor scheme. In another experiment, the changes in the image quality for different levels of embedding capacity are investigated in the *n*-predictor policies and the results are compared with all the competing schemes in Figure 4.12(c) and Figure 4.12(d). The figures demonstrate that the schemes loss the SSIM value for the achievement of every larger embedding capacity. The figures also state that the proposed multi-predictor policies maintain the higher value of similarity index and the similarity index value is larger in the scheme that

applies more predictors. This implies that the proposed scheme manages better stego image quality.

From these analyses on the PSNR and the SSIM values, it is concluded that the *n*-predictor policies preserve better stego image quality and embed more quantity of bits for the same amount of image distortions in the CalTech image dataset.

4.4.4 Performance Evaluation of the Schemes in Different Standard Image Datasets

The *n*-predictor and the reviewed schemes are experimented in other image datasets as well. The same data stream is embedded into each of the images by all the stated reviewed schemes and the proposed one. The results are presented in this subsection by averaging the values of payloads, PSNRs and SSIMs for each image dataset separately. This subsection also analyses the minimum and the maximum embedding gains in each of the image datasets for each of the schemes. Finally, the performances of *n*-prediction policies are presented for analyzing themselves.

4.4.4.1 Analysis of Embedding Payloads in Diverse Image Datasets

The average payloads of all the SPP and the BPP based schemes, obtained in different image datasets, are tabulated in Tables 4.5 and Table 4.6, respectively. The amount of payloads obtained by different schemes varies between the image datasets. The CalTech and the satellite images provide higher embedding payloads while the texture images present smallest payloads in all the experimented schemes. The reason is that most of the CalTech images highlight a single object. Variations among the neighbor pixels in such single object highlighting frames are small in magnitude. This virtue helps the predictors to predict more accurately and to enhance the quantity of the embeddable errors. Again, the satellite images contain cloud like large gray regions where the pixel values are very close to each other. Consequently, the frequencies of embeddable errors of −1 and 0 rise significantly in these two categories of images. On the other hand, in the texture images, greater transitions in pixel values between the pixels of each two neighbor regions are observed; whereas the natural and the standard images contain random pixel variations. Hence, the payloads in the images of these two last categories are at the moderate level.

Table 4.5: Average payloads in the SPP based schemes in different image datasets

Schemes	CalTech101	Standard	Natural	Texture	Satellite
Tai	15409	10362	9064	2891	12231
Wien	16011	10615	9324	3256	12247
Weijen	9735	4710	5152	987	9834
Xiao2	18067	13458	11808	4112	13980
Xiao3	18880	14477	12903	4511	14533
Xiao4	18282	13770	12114	4248	14130
2-predictor	17739	13017	11243	4419	13760
3-predictor	19519	15301	12917	5756	15170
4-predictor	20641	16785	14354	6421	16210
5-predictor	22780	19678	16814	8080	18146

Table 4.6: Average payloads in the BPP based schemes in different image datasets

Schemes	CalTech101	Standard	Natural	Texture	Satellite
Kamal	18190	13759	13173	7350	15796
Leung	18944	14477	13106	7805	16302
Wien	18293	13902	13595	7351	15708
Tsai	18163	13516	12494	7370	15806
Mid	11525	7568	6933	1732	9825
P2:KL	25376	22643	20652	14690	22643
P3:KLW	27142	24776	23212	16348	24617
P4:KLWT	30771	29591	27609	20869	28884
P5:KLWTM	30099	29161	27028	20454	28169
Truncation	1051	1669	782	424	1463

While comparing embedding payloads among the SPP based schemes, it is investigated that the multi-predictor based methods, i.e., Xiao2, Xiao3, Xiao4, 2-predictor, 3-predictor, 4-predictor and 5-predictor, present higher payloads. Among the multi-predictor based methods, the proposed *n*-prediction policies exhibit their superior performance on presenting better payloads. The results are tabulated in Table 4.5. Similar experiments are performed for the BPP based schemes and their results are put in Table 4.6. The average payloads tabulated in Table 4.6 underpin that the *n*-predictor policy notably dominates the competing schemes.

Again, in both the SPP and the BPP based *n*-predictor policies, this is also investigated that the scheme employed more predictors provides higher embedding payloads in all the image datasets.

Table 4.7: Payload gain of 3-predictor scheme over the others SPP schemes in diverse image datasets

Gain of 3-predictor	**CalTech101**		**Standard**		**Natural**		**Texture**		**Satellite**	
over the ↓	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**
Tai	24	123	18	329	19	193	44	418	17	136
Wien	19	120	15	191	20	189	41	202	17	138
Weijen	17	463	15	627	19	529	73	460	10	392
Xiao2	17	55	12	81	18	88	32	103	18	57
Xiao3	19	42	17	63	14	57	23	68	17	37
Xiao4	14	66	18	72	13	75	30	91	17	49

Table 4.8: Payload gain of 3-predictor scheme over the others BPP schemes in diverse image datasets

Gain of 3-predictor	**CalTech101**		**Standard**		**Natural**		**Texture**		**Satellite**	
over the ↓	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**
Kamal	22	136	11	158	20	165	78	160	12	145
Leung	15	127	19	153	17	149	59	152	19	138
Wien	30	146	18	169	27	175	84	174	19	159
Tsai	23	158	14	182	24	183	86	183	16	165
Mid	47	1018	34	2475	41	1536	185	2930	27	1141
Truncation	1197	6762	471	6705	790	7093	1018	9233	763	5666

The results are also analyzed in terms of payload gains. The payload gains of the proposed *n*-predictor policies are always positive and this is a few multiple of some of the competing ones. As a sample, payload gains in 3-predictor policy regarding each of its competing schemes are presented in Table 4.7 and Table 4.8 for each image dataset, respectively, for the SPP and the BPP based techniques. Table 4.7 states that the minimum and the maximum gains are 17% and 418% over Tai, 15% and 202% over Wien, 10% and 627% over Weijen, 12% and 103% over Xiao2, 14% and 68% over Xiao3, 13% and 91% over Xiao4. Table 4.8 affirms that the minimum and the maximum gains are 11% and 165% on Kamal, 15% and 153% on Leung, 18% and 175% on Wien, 14% and 183% on Tsai, 27% and 2930% on Mid, 471% and 9233% on Truncation. These afore gain figures validate that the

proposed scheme outperforms in all the image datasets regarding the volume of data implantation.

4.4.4.2 Analysis of PSNRs in Diverse Image Datasets

An average of PSNR values for each dataset is tabulated in Table 4.9 and Table 4.10 separately for the SPP and the BPP techniques, respectively. Both the tables demonstrate that the PSNR values are higher in the proposed *n*-predictor policies. The PSNR of 50dBm or more is observed only in the proposed schemes. These PSNR values validate that the proposed scheme preserves the image quality better than the others. The reason is that the proposed predictor generates more quantity of embeddable errors. This is realizable from the Eq. 4.2 that the error value, and finally the pixel value, remains unchanged for embedding a bit of value 0. Consequently, the improved number of embeddable errors by the *n*-predictor policies has increased the number of unaltered pixels in the stego image regarding its cover pixels. As a result, the proposed scheme demonstrates better PSNR values.

4.4.4.3 Analysis of SSIM Values in Diverse Image Datasets

A larger value of structural similarity index indicates better similarities between two compared images. Table 4.11 and Table 4.12 show the SSIM values, respectively, in the SPP and the BPP techniques. Table 4.11 demonstrates that the values of SSIM obtained in the *n*-predictor policies are either higher or much closed to others. Table 4.12 outlines a definite improvement in SSIM values by the proposed scheme.

4.4.4.4 Analyzing the Effect of Increasing the Number of Predictors in the *n*-Predictor Scheme

The number of applied predictors in the *n*-predictor scheme has impacts on the time complexity, the payload and the image quality. The time complexity of the *n*-predictor scheme is proportionate to the number of predictors used because each predictor predicts pixels separately. If the quantity of pixels in a cover image is t, then the time complexity of the *n*-predictor scheme to predict the cover pixel values will be $O(nt)$, while it is $O(t)$ for a single predictor based scheme. The time complexity of embedding data is always $O(t)$, regardless the number of predictors used. Hence, the total time complexity of the *n*-predictor scheme is $O(nt + t)$, while it is $O(2t)$ for the other schemes. The notation $O(nt + t)$ is not a polynomial

representation and, thus, the recent computer may complete the whole process within a time ranges from very few seconds to a minute depending on the size of image and message length. In most embedding applications, the issue of improving the payloads and the image quality by a scheme is more important than its execution times if the required time is not too long.

Table 4.9: Average PSNRs in the SPP based schemes

Schemes	CalTech	Standard	Natural	Texture	Satellite
Tai	49.412	49.039	48.918	48.453	49.215
Wien	49.467	49.071	48.941	48.478	49.216
Weijen	49.442	49.109	49.090	48.457	49.320
Xiao2	49.698	49.362	49.206	48.563	49.385
Xiao3	49.714	49.381	49.234	48.566	49.393
Xiao4	49.660	49.322	49.167	48.546	49.361
2-predictor	49.854	49.556	49.365	48.677	49.524
3-predictor	49.979	49.705	49.476	48.767	49.621
4-predictor	50.002	49.735	49.509	48.772	49.634
5-predictor	50.145	49.921	49.651	48.869	49.754

Table 4.10: Average PSNRs in the BPP based schemes

Schemes	CalTech101	Standard	Natural	Texture	Satellite
Kamal	49.466	49.131	49.074	48.619	49.318
Leung	49.532	49.192	49.068	48.650	49.366
Wien	49.475	49.143	49.110	48.619	49.309
Tsai	49.467	49.115	49.020	48.620	49.326
Mid	48.935	48.667	48.613	48.241	48.854
P2:KL	50.289	50.085	49.935	49.306	50.077
P3:KLW	50.289	50.085	49.935	49.306	50.077
P4:KLWT	50.675	50.572	50.362	49.700	50.495
P5:KLWTM	50.604	50.530	50.303	49.664	50.422
Truncation	0.380	0.363	0.387	0.397	0.369

When the proposed scheme uses more predictors in its prediction phase, the quantity of embeddable errors is increased. This means that the embedding phase implants more message

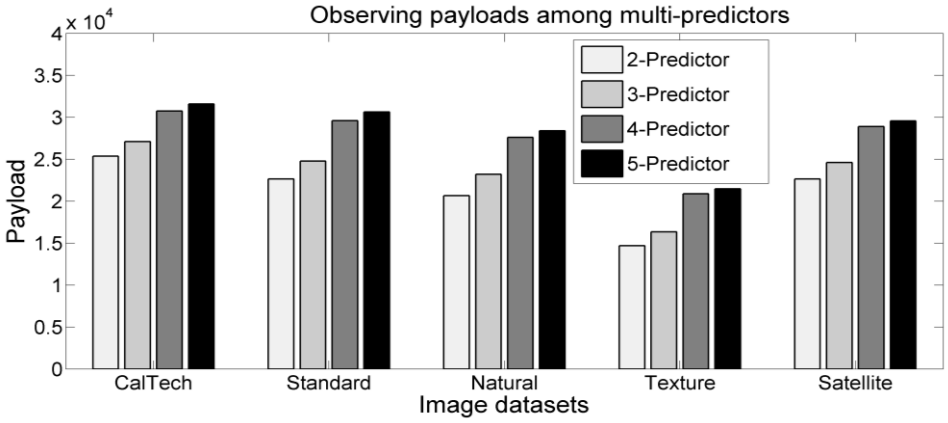
bits. As the embedding rules do not change the value of an error as well as the respective pixel value while implanting a message bit 0, the quantity of unaltered pixels is increased. Due to producing more embeddable errors, the values of the payload, the PSNR and the SSIM are improved in the proposed scheme. The effects on the values of the payload, the PSNR and the SSIM due to the use of multiple predictors are demonstrated in Figure 4.13 (a)–(c) respectively. In these figures, the average value computed for each image dataset is demonstrated as a bar chart. All these figures justify that the better values for the payload, the PSNR and the SSIM are found for the employment of more number of predictors.

Table 4.11: Average SSIMs in the SPP based schemes

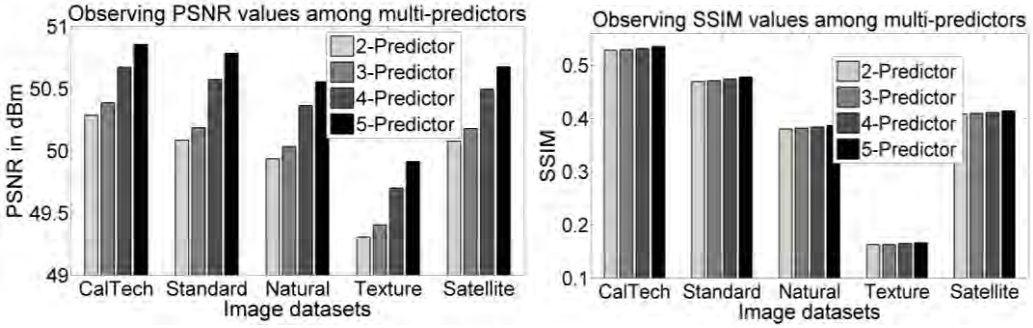
Schemes	CalTech101	Standard	Natural	Texture	Satellite
Tai	0.493	0.450	0.360	0.157	0.389
Wien	0.495	0.452	0.361	0.158	0.390
Weijen	0.489	0.445	0.357	0.155	0.386
Xiao2	0.493	0.450	0.360	0.156	0.388
Xiao3	0.494	0.451	0.360	0.157	0.389
Xiao4	0.494	0.451	0.360	0.157	0.389
2-predictor	0.492	0.448	0.358	0.156	0.387
3-predictor	0.493	0.449	0.359	0.156	0.388
4-predictor	0.493	0.449	0.359	0.156	0.388
5-predictor	0.494	0.451	0.360	0.157	0.389

Table 4.12: Average SSIMs in the BPP based schemes

Schemes	CalTech101	Standard	Natural	Texture	Satellite
Kamal	0.520	0.459	0.372	0.157	0.400
Leung	0.521	0.461	0.373	0.159	0.403
Wien	0.527	0.467	0.380	0.162	0.408
Tsai	0.524	0.464	0.375	0.161	0.405
Mid	0.527	0.467	0.378	0.162	0.405
P2:KL	0.528	0.469	0.380	0.163	0.408
P3:KLW	0.529	0.470	0.381	0.163	0.409
P4:KLWT	0.532	0.474	0.384	0.165	0.411
P5:KLWTM	0.533	0.475	0.385	0.166	0.412
Truncation	0.034	0.024	0.040	0.137	0.202



(a)



(b) (c)

Figure 4.13: Observing the effects of using more predictors in the *n*-predictor scheme on performance measuring parameters on diverse image dataset: (a) payloads; (b) PSNR; and (c) SSIM.

Among the image datasets, the scheme shows the less performance in the texture images because of having large valued gradients, i.e., the difference between the values of two neighbor pixels are observed due to the existing frequently sudden transitions. The poor performance of the predictors increases the quantity of non-embeddable errors. Non-embeddable errors are shifted by an amount during application of the embedding rules and thus, destroy the image quality and decrease the value of the payload. Nevertheless, the analyses of this sub-section conclude that the *n*-predictor scheme shows superior performance regarding all the performance measuring parameters independently in the image datasets and the performance increases for employment of more predictors.

4.4.5 Resistance to Statistical Attacks

Image steganography shields the disclosure of the secret data communication by hiding them into a cover image and makes the embedded data visually insensible. The *n* predictors and their application sequence defined by a negotiation between two communication parties impose five steps securities. Firstly, the third party does not know the number of predictors that were applied by the encoder. Secondly, without knowing the exact predictors A_k, the decoding is not possible. Thirdly, if the lists of parameters of the predictors are not known and the parameters are not initialized with proper values, no one can predict accurately. Fourthly, the values of $B_{n,m}$, $\beta_{n,m}$ and *m* in the Eq. (4.8) will protect secret data from all unauthorized decoders, as it cannot generate the hybrid errors without them. Fifthly, without generating exact *AP* one cannot presume an optimal prediction error. To generate *AP*, one should know the exact sequence of the applied predictors along with other parameters. Though it is very hard to break these series of data securities, in a random fashion, one can deploy statistical analysis to comprehend the existence of the secret information within the stego image. To justify the steadfastness of the proposed scheme against statistical attacks two famous and latest techniques are analyzed in this chapter. These techniques work by analyzing the histograms of differences of adjacent pixels (HDAP) in the stego image and the behavior of cover and stego pixels by the generalized Benford's Law (gBL). The HDAP and gBL methods are described in Section 2.7 in details.

4.4.5.1 Testing the Security of the Proposed Scheme by the HDAP Method

For analyzing the HDAP method, adjacency pixel differences along the vertical and horizontal directions are computed. Four the snapshots of these two histograms are depicted in Figure 4.14. Each snapshot, comprises of both vertical and horizontal difference histograms, displays no anomalies between the vertical difference histogram, $\tilde{H}_v$ and the horizontal difference histogram, $\tilde{H}_h$. A statistical difference *D* between $\tilde{H}_v$ and $\tilde{H}_h$ is measured. The value of *D* becomes large when more quantities of pixel values are modified by the embedding rules. The values of *D* are plotted in Figure 4.15 for sixty images. It is observed that the proposed scheme comprises of two predictors provides the best performance as its *D* values are smaller than the others. It happens due to the fact that the proposed scheme increases the number of embeddable errors and thus, enhances the frequency of unchanged pixels.

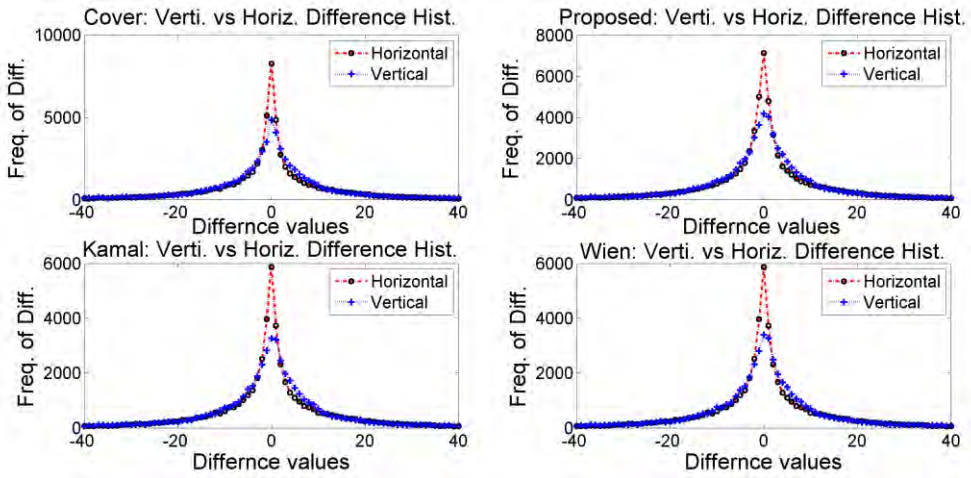


Figure 4.14: Vertical and Horizontal Difference Histograms

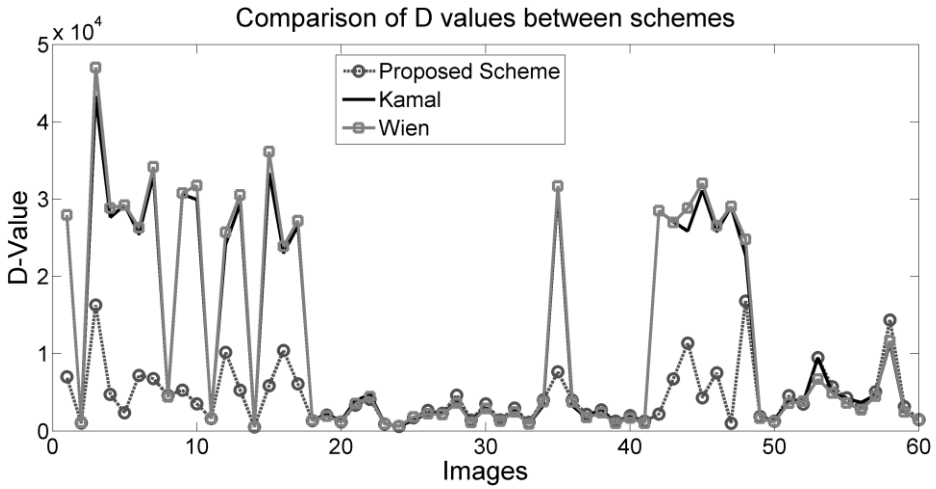


Figure 4.15: *D*-values between schemes.

4.4.5.2 Testing the Security by the gBL

The proposed scheme measures the values of d_i, stated in Eq. (2.18), and analyses their values. As a sample, the results of first four digits, i.e. 1 to 4, which are measured in our experiment, are presented in Figure 4.16, where the y-axis represents the values of d_i and each minor tic along x-axis represents an individual image. The values are very small. For the convenience of visualization, the results of the proposed scheme and the Hong *et al.*'s (2010) scheme are compared. From the figure, it is observed that the d_i values are smaller and very close to zero in the proposed scheme. This indicates that the changes in the digits are negligible. Hence, it can be concluded that the proposed embedding scheme has enough resistance to prove its effectiveness against any statistical attacks.

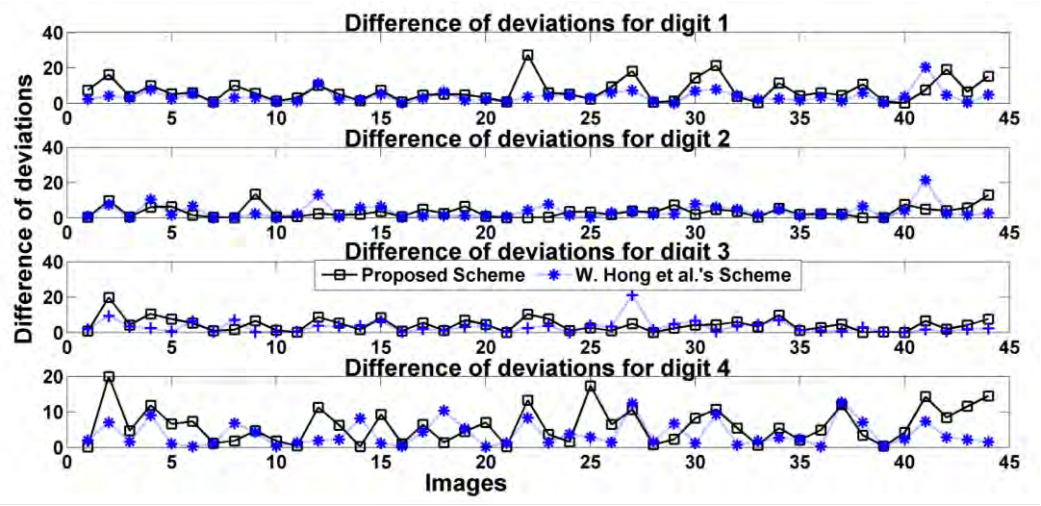


Figure 4.16: Result of Generalized Benford's Law in the digits 1-4.

4.5 Conclusions and Summary

The reversible schemes suffer from the lower embedding capacity. Nevertheless, many applications related to forensic, medical, military and law enforcing agencies utilize both the extracted secrets and the retrieved cover image to their further processing stages at their decoder end. Increasing the embedding capacity and enhancing the stego image quality are, therefore, an attractive area to the researchers. The proposed *n*-predictor scheme increases the embedding capacity notably. The scheme is experimented on five different image datasets. These reveal that it enhances the embedding capacity of the SPP based techniques by 10%–627% and the BPP based techniques by 11%–9233% depending on the texture properties and the pixel variations in the images. The self-reliant decoder can extract the secret message and also retrieve the cover image from the stego one. It is an effective method to meet the demand for larger embedding capacity. The analyses on both the PSNR and the SSIM ensure that the proposed scheme minimizes the distortions in the stego image compared to its competing schemes. The use of multiple predictors increases the security of the scheme as the list of the applied predictors, the prediction parameters and their application sequences are not open to any third party. The scheme also demonstrates its stronger resistance capability against the known statistical attacks.

Chapter 5

Multilayer Multi-cycle Embedment Process

In this chapter, a multilayer multi-cycle data embedment process is proposed, where embedment of secret bits is performed into the errors of multiple layers for multiple times in the prediction error histogram. Prediction error based traditional multilayer data embedment schemes conceal secret bits into a number of distinct and contiguous high-frequency errors. In an n layer data embedment scheme, concealment of data is performed in zero and its n neighbor errors from each of the positive and the negative sides, i.e., $2n$+1 errors. Findings show that the value of the better part of the prediction errors is very close to zero and the data embedment capacity and the image quality drop sharply if the embedding is performed in a wider range of embeddable errors, i.e., for a large value of n. Hence, most of the traditional schemes restrain themselves from embedding data into a wider range of embedding layers. The investigation also states that the data embedment for k-times into the n/k layers of the prediction errors produces higher embedding payloads and maintains better stego-image quality compared with the n layer data embedment scheme concealing data for a single time only. The proposed multilayer multi-cycle scheme explores the points at which significantly better payloads can be obtained while maintaining a minimal image distortion. An improved performance in terms of embedding capacity and stego image quality was obtained substantially during empirical analysis, especially in the scenario of embedding large volume of data.

5.1 Introduction

In the prediction error based reversible data embedment schemes, data implantation is performed into a certain prediction error values [31, 32, 113, 114] known as the embeddable errors. The embeddable errors are defined from the contiguous peak-presented errors in a prediction error histogram (PEH). Embedding is performed into one or two peak-presenting errors in a single layer data embedment scheme. It is investigated that the highest appeared

error is always zero ('0') as shown in Figure 5.1. In most cases, the next highest appeared error is -1. Hence, in a single layer data embedment scheme, also known as the layer 0 data embedment scheme, the hidden message is implanted either in the error value of 0 or in both 0 and -1. Though the single layer data embedment schemes ensure smaller degradation in the stego image quality, these schemes suffer from smaller embedding capacity.

Many applications demand large embedding capacity and the size of the data to be embedded is increasing day by day. In the recent literature, multiple high-frequency layers of errors in the PEH are used to embed data in order to get better embedding capacity [31, 51, 54, 84, 113, 114]. In an L layer data embedment scheme, for $L \geq 0$, message implantation is performed into the errors whose values range from $-L$ to $+L$, i.e., into $2L+1$ dissimilar embeddable errors. The Figure 5.1 demonstrates the definition of layers in the PEH. Arrow lines from a layer label indicate the range of the positive and the negative outer errors that are encompassed within that layer. Although it is demonstrated in the Figure 5.1 that the layer 0 encompasses only one error, called zero, in practice, it incorporates two errors, namely -1 and 0. Embedding secret message in layer 1 ($L=1$) means that message bits are concealed into the prediction errors of -1, 0 and 1 only, i.e., into 3 errors. Similarly, a scheme using 2-layer embedment ($L=2$) implies that the message bits are implanted into the error values of -2, -1, 0, 1 and 2, i.e., into 5 errors.

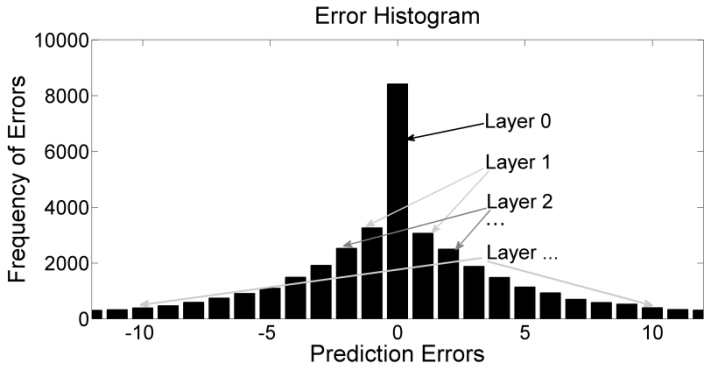


Figure 5.1: Different layers of errors in a prediction error histogram.

The value of employed error layers in the multilayer data embedment schemes depends on the volume of data to be embedded. During the embedment of large volumes of data, more error layers are employed. Nevertheless, using a higher number of layers ultimately results in a smaller increase in the embedding capacity because the frequency of each error decreases sharply from that of its immediate smaller valued error (in their absolute magnitude), as shown

in the Figure 5.1. On the contrary, employment of a large number of embedding layers, indeed, increases the amount of error shifting and hence, decreases the peak-signal-to-noise-ratio (PSNR) as well as the structural similarity index matrix (SSIM) values of the stego image. Therefore, rather than embedding data into the errors in an L layer data embedment process for a single time, embedding data bits into the L/k layer for k-times will provide larger embedding payload and better stego image quality.

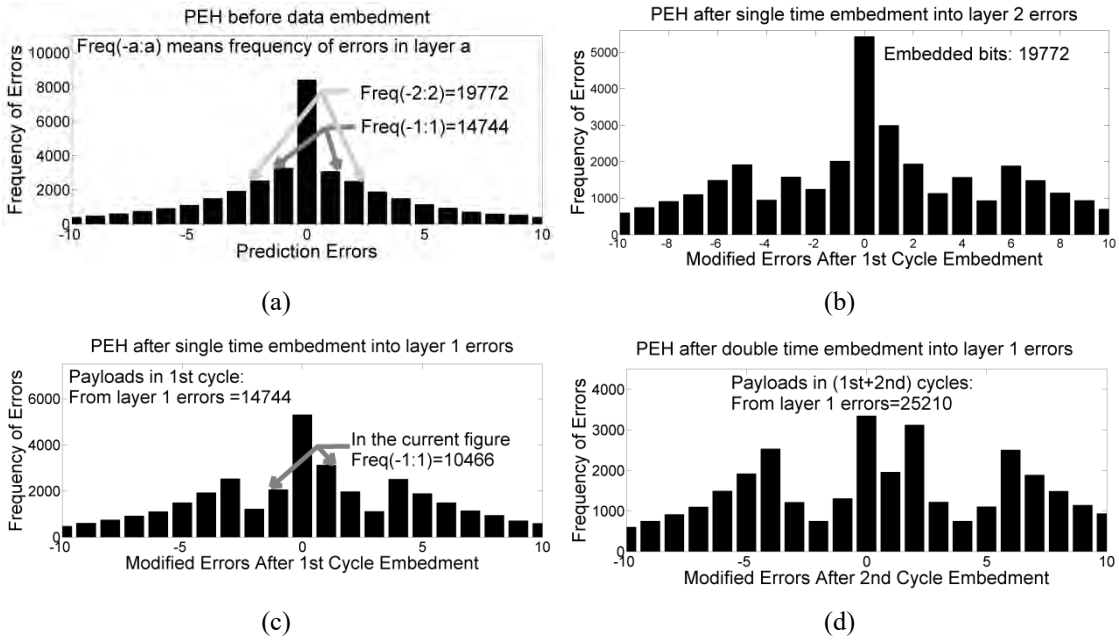


Figure 5.2: Prediction error histogram of Lena image: (a) before data embedment; (b) after single cycle data embedment into layer 2 errors; (c) after single cycle data embedment into layer 1 errors; and (d) after double cycle data embedment into the errors of layer 1.

Figure 5.2 demonstrates the justification of obtaining better payloads and PSNR values from the use of L/k embedding layers for k-times compared with the one from the use of a single cycle L layer data embedment scheme. For the simplicity, the demonstration is performed for $L = 2$ and $k = 2$. A PEH generated by applying a block median predictor on the Lena image of size 255×255 is shown in Figure 5.2 (a). The figure demonstrates that the number of errors with value ranges from -1 to 1 is Freq(-1:1) = 14744, while Freq(-2:2) = 19772. If a scheme A_1 embeds message bits into every error of L=2 of the computed prediction errors, as shown in Figure 5.2 (a), it yields the payloads of 19772bits and the change of PEH is depicted in Figure 5.2(b). Alternatively, the scheme A_2 may implant message bits for double times into every error of layer L_D=L/2. Embedding into the errors of layer L_D=1 for the first

time, the scheme yields a payload of 14744 bits. The modified PEH, due to data embedment into the errors of L_D =1 for a single time, is depicted in Figure 5.2(c). After the first cycle, if the scheme A_2 is allowed to embed again, i.e., for second times, into the errors of L=1 of the Figure 5.2(c), the scheme will embed 10466 bits more. The modified PEH after the double cycle data embedment by the scheme A_2 is presented in Figure 5.2(d). After the double cycle data embedment process, the scheme A_2 will present 25300 bits of payloads in total. This payload is greater than the payload of L=2 single cycle data embedment, i.e., 19772bits. Thus, the demonstration proves that the double cycle process produces higher payloads by embedding into the errors of layer 1 compared with the scheme that embeds into the errors of layer 2 for a single cycle. In a constraint of achieving the same amount of embedding payloads, the double cycle scheme will present better stego image quality than the single cycle process. To achieve the same payloads, i.e., 25300 bits by a multilayer single cycle scheme, it has to embed into the errors of L=4, i.e., error points of {-4, -3, -2, -1, 0, 1, 2, 3, 4}, because Freq (-3:3)=23580 and Freq(-4:4)=26556 in the Figure 5.2(a). In case of applying single cycle embedment process into the errors of 4 layers, the positive and the negative valued non-embeddable errors have to be shifted by 4 and 5 units respectively to prepare enough space for the movement of embeddable errors, whereas in the 1-layer double cycle scheme, these negative and positive valued non-embeddable errors have to be shifted by 2 and 4 units respectively. The shifting of pixel values is deeply affected by the layer value. Hence, a scheme will introduce higher order of image distortions by exploiting errors of 4 layers in their single cycle process rather than errors of 1 layer in a double cycle process. The scenario will be more promising (both for payload and PSNR) if the value of L is replaced by a higher magnitude and k is chosen to a larger value than 2. In addition, attackers may classify the highly distorted stego images by employing statistical attacks. Consequently, increasing the embedding payloads by employing more layers in the single cycle scheme may introduce a new dilemma of successful statistical attacks. Hence, in addition to improving the image quality, the use of multi-cycles will introduce new security features like the number of exercised cycles and quantities of exploited embeddable error points which are part of a shared key. These analyses motivate to propose the thesis in favor of using L/k embedding layers for k-times rather than L layers for a single time.

In the present chapter, the proposed scheme implements a novel multilayer, multi-cycle (MLMC) embedding scheme by applying the same embedding process for multiple rounds into the errors of a defined layer. The proposed scheme verifies that the multi-cycle embedding attempt enhances the embedding capability of the scheme to hide massive amounts of data.

The process of embedding hybrid data of type numeric, text and audio is outlined in this scheme. This unique representation will expand the scope of the scheme to be utilized for various purposes like hiding text and audio data related to interviews, investigations and reports; and such in forensic, medical, scientific and many other official applications. The estimation process for an optimal layer value is presented in this multi-cycle scheme. The scheme also verifies that to achieve the same capacity of k cycle embedding into L/k layers, i.e., MLMC, the multilayer, single cycle (MLSC) scheme has to entrench in more than L layers. This implies that MLMC scheme reduces the total amount of error shifting. To justify the claims, experiments are conducted on diverse image datasets - CalTech dataset comprising of 5000 images, BOSS dataset contained 500 images and a standard dataset of 50 images. The experimental results and analyses prove that the multi-cycle embedding scheme successfully implants massive and hybrid data of type numeric, text, and audio. The MLMC scheme provides both higher embedding capacity and stego image quality simultaneously.

The rest of the chapter is organized into five more sections. In Section 5.2, list of reviewed schemes is provided. Section 5.3 is devoted to detailing the proposed multi-cycle embedding process. In Section 5.4, the experimental setup, results and their discussions are demonstrated. The resistance against statistical attacks is tested in Section 5.5. Finally, Section 5.6 provides concluding remarks.

5.2 State-of-the-Arts

Tai *et al*. in 2009 [83] and Zhao *et al.* in 2011 [113] applied differences of adjacent pixel pairs as an embedding space in their multi-layer embedding schemes. Luo *et al.* in 2011 [61] produced a difference histogram by exploiting spatial correlation among block pixels. Embedding capacity for these schemes is still very low as they embed bits by modifying the histogram of adjacency pixel differences, known as the spatial errors, and it is evident that the frequencies of several peak errors in the PEH are much higher than those in the spatial error histogram. Therefore, many schemes, e.g., Hong in 2012 [31], Hong *et al.* in 2010 [32], Kamal and Islam in 2015 [40], Leung *et al.* in 2013 [51], Govind *et al.* in 2015 [26] and Wang *et al.* in 2014 [94], utilize the prediction errors as an embedding space to achieve higher embedding capacity. Though the schemes exhibit their variations in their prediction methods and some processing stages, these schemes behave just the same during their data embedment. Pan *et al*. in 2015 [114] introduced a different approach of embedding into spatial domain by using

multi-layer embedding process. All of these stated schemes are briefly described in Chapter 2 along with their limitations.

5.3 Proposed Multilayer Multi-Cycle Embedment Scheme

The proposed scheme implants message bits into the prediction errors generated by a predictor. It does not enhance the prediction accuracy of the applied predictor, rather improves the embedding capacity by implanting bits for multiple times into the errors. Therefore, what predictor is applied in the prediction phase of the scheme is not a concerned issue of this Section. Let $e_{i,j}$ is the measured prediction error of the pixel at location (i, j) in the image I. The error values $e_{i,j}$, where $-255 \le e_{i,j} \le 255$, are termed as the cover errors. The scheme first defines the multi-cycle embedding layer (*MC_L)* in the prediction error histogram. The embedding layer *MC_L* comprises of the errors from -*MC_L* to *MC_L*. These error samples are known as the primitive embeddable errors. Primitive embeddable errors are measured at stage zero, i.e., before starting the data embedment task. In the first round, the proposed multi-cycle scheme embeds secret bits into the errors of *MC_L* layers of the PEH, i.e. into the error points of $-MC\_L$, ..., -1, 0, 1, ..., MC_L, by the embedding rules of [51]. An error point represents a sample value and the frequency of an error point could be greater than or equal to zero. After this round, the processed errors are termed as the stego errors at stage one. It is investigated that many of the errors of stage one still hold their primitive values, i.e., many embeddable errors of stage zero remain unchanged at stage one. Even, the majority of the primitive embeddable errors of stage zero do not cross the range [-*MC_L, MC_L*] after the first round of data implantation, as shown in Figure 5.2(c). The stego errors of first embedding cycle, i.e., errors at stage one, are reused for data embedment in the second cycle. The second cycle employs the same embedding procedure to implant data bits into the errors of stage one, i.e., into the error values of $-MC\_L$, ..., -1, 0, 1, ... and MC_L. The second round generates the stage two stego errors. The process repeats its task for k times in a k-cycle embedding scheme. The flowchart depicted in Figure 5.3 portrays the process. In this block diagram, it is observed that the scheme starts for $c = 1$ and embeds data bits into the errors of $e_{i,j}$. After the completion of bit implantation into the errors for a round, the cycle counter c is increased by 1. If the execution cycle c is not greater than the planned cycle k, it assigns the modified error $\tilde{e}_{i,j}$ in $e_{i,j}$ to reuse the errors by the same embedding rules in the next round.

This way, the flowchart depicts the module that embeds data bits into the errors for *k*-times and finally the module supplies the stego errors to its next phase for the generation of stego pixels.

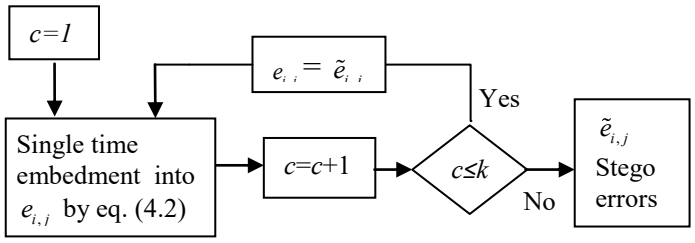


Figure 5.3: Flowchart of multi-cycle embedding scheme

5.3.1 Defining Embeddable Errors in MLMC Schemes

The MLMC scheme employs more than one layer, i.e., $ML\_C > 0$, for *k*-times to embed data bits in its *k*-cycle scheme. That is, as embeddable errors, the scheme employs $(2MC\_L+1)$ separate valued error points in each of the *k* cycles of the embedding process. In each of the stage, the scheme encounters $(2MC\_L+1)$ different error points. Thus, the process encounters a total of $k(2MC\_L+1)$ error points during its *k*-cycles. To relate the schemes of MLSC and MLMC for the measurement of embedding efficiency, total encountered error points $(2L+1)$ in MLSC and $k(2MC\_L+1)$ in MLMC should be theee same and hence, $(2L+1) = k(2MC\_L+1)$, where L is the embedding layer of MLSC scheme. As k, L and MC_L are integer values, $(2L+1)$ and $k(2MC\_L+1)$ will not be equal for all the values of k, L and MC_L, e.g. if L=4 then both $(2L+1)$ and $k(2MC\_L+1)$ will be equal only for k=3, MC_L=1 and if L=7 then the equality will be held only for k=5, MC_L=1 or k=3, MC_L=2. This means that if k is allowed to be chosen to any value, the relation $(2L+1) = k(2MC\_L+1)$, i.e., $2MC\_L+1 = (2L+1)/k$, will not be held always, e.g., the equality in the stated relation will not be held for L=7 and k=4. In that case, total encountered errors in both the MLSC and the MLMC schemes are managed to be very close valued. This is done by the relation $(2MC\_L+1) = (l+k-\text{mod}(l,k))/k$, where $l = (2L+1)$ and $\text{mod}(l,k)$ is the modulus value of l and k. Then, $(l+k-\text{mod}(l,k))/k$ can be either an odd or an even number. Let, $l' = (l+k-\text{mod}(l,k))/k$. The negative valued embeddable error points in

MLMC scheme will be from -1 to $-\left\lfloor \frac{l'}{2} \right\rfloor$ (i.e., total $\left\lfloor \frac{l'}{2} \right\rfloor$ different errors) whereas the positive-valued embeddable error points will be either from 0 to $\left\lfloor \frac{l'}{2} \right\rfloor$-1 or from 0 to $\left\lfloor \frac{l'}{2} \right\rfloor$ (i.e., $\left\lfloor \frac{l'}{2} \right\rfloor$ or $\left\lfloor \frac{l'}{2} \right\rfloor$+1 different errors) depending on whether l' is even or odd valued respectively. Hence, the proposed scheme further replaces the concept of the *MC_L* by two parameters – number of different negative valued embeddable error points (nNEP) and number of different positive valued embeddable error points (nPEP), while, the negative valued embeddable error points are from -1 to -*nNEP* and positive valued embeddable error points are from 0 to *nPEP*-1, as shown in Figure 5.4 by two separate gray colors. In the figure, the bars with bright gray color, mid-level gray color and black color represent the positive-valued embeddable errors, negative valued embeddable errors and non-embeddable errors, respectively. When $\text{mod}(l,k)=0$, the MLMC scheme sets $nNEP=l'/2$, $nPEP=l'/2$; otherwise it sets $nPEP=\left\lfloor \frac{l'}{2} \right\rfloor+1$ and $nNEP=\left\lfloor \frac{l'}{2} \right\rfloor$.

The replacement of *MC_L* by *nNEP* and *nPEP* will allow the scheme to extend its scope of selecting an arbitrary number of embeddable error points in a side, depending on different types of histogram properties and demand of the applications.

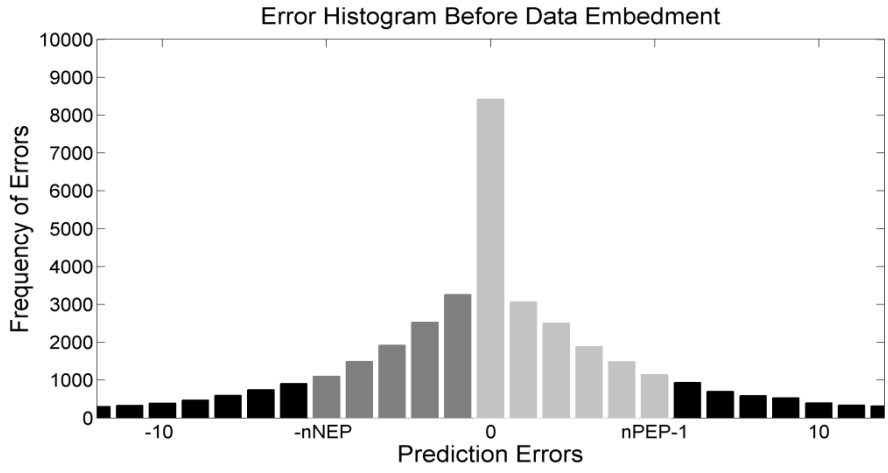


Figure 5.4: Categorizing embeddable error points.

5.3.2 Defining the Data Embedment Principles in MLMC Schemes

Each message bit *m* of the message stream *M* is implanted into an embeddable error of $e_{i,j}$, say, err, where $-nNEP \le err \le (nPEP-1)$ using the embedding rule given in Eq. (5.1). The equation modifies the negative valued non-embeddable errors by *-nNEP* and the positive-valued non-embeddable errors by *nPEP* by the first two rules of the Eq. (5.1), respectively. The third rule of the Eq. (5.1) implants data bits *m* into an error $e_{i,j}$ when $e_{i,j}$ becomes an embeddable error. The embeddable errors are modified by $2e_{i,j}$ or $2e_{i,j}+1$ depending on implanted bit 0 and 1 respectively. These modified errors are $\tilde{e}$.

$$\tilde{e}_{i,j} = \begin{cases} e_{i,j} - nNEP & if \;\; e_{i,j} < -nNEP \\ e_{i,j} + nPEP & if \;\; e_{i,j} \ge nPEP \\ 2e_{i,j} + m & Otherwise \end{cases} \qquad (5.1)$$

Algorithm 5.1*: kCycleEmbedment (* $e_{i,j}$ *, M, k, nNEP, nPEP)*

Step 1. *Set cycle=1,* $tBits = |M|$

Step 2. *Read a bit m from M. Embed the m into* $e_{i,j}$ *by the Eq.*(5.1)

Step 3. $tBits = tBits - 1$

Step 4. *If* $tBits \neq 0$ *and all errors in* $e_{i,j}$ *are not tried, go to Step* 2

Step 5. *Set cycle=cycle+1*

Step 6. *If cycle≤k and* $tBits \neq 0$, *set* $e_{i,j} = \tilde{e}_{i,j}$ *and go to step* 2

Else Go to step 7

End if

Step 7. *Announce the completion and stop the task*

Figure 5.5: k cycle data embedment algorithm.

The MLMC scheme applies the Eq. (5.1) in its each embedding cycle to implant data bits. At each cycle, the scheme implants the bits of the bitstream *M* into *nPEP+nNEP* error points, i.e., error values range from *-nPEP* to *nNEP*-1. The seven steps of Algorithm 5.1, i.e., steps 1 to 7, in Figure 5.5 define the *k*-cycle embedment process. This *k*-cycle process embeds total |*M*| bits, say *tBits*, into the errors, where |.| stands for the length of a string. The algorithmic step 1 of the Algorithm 5.1 initializes values for the execution controllable parameters. Steps 2

to 4 implants bits in a cycle by the Eq (5.1). Steps 5-6 are used to repeat the cycles for *k* times. Finally, step 7 terminates the execution task.

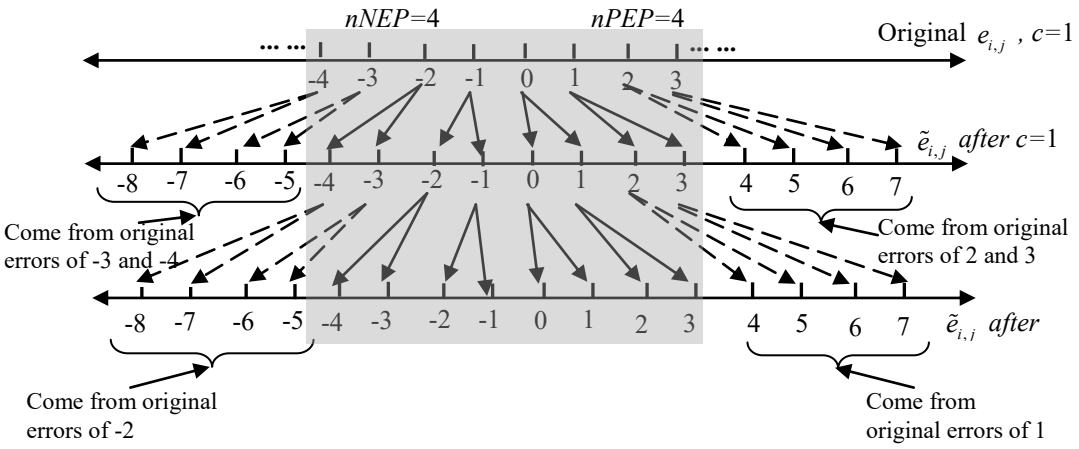


Figure 5.6: Error shifting by first and second cycle data embedment process.

For an example, Figure 5.6 depicts a 2-cycle embedding scenario where both the *nNEP* and *nPEP* are set to 4. The figure narrates the functionality of the Eq. (5.1) in each embedding cycle. Errors of each error points are modified by two stego errors in each embedding cycle by using the Eq. (5.1), e.g., errors of values of -1 and 0 are modified to the values of -2, -1 and 0, 1 respectively, depending on the message bit m to be implanted in each error. In the figure, the primitive embeddable errors $e_{i,j}$ are labeled just below the first horizontal line. These are the embeddable errors at stage zero. The stego errors at stage one are provided just below the second horizontal line. Similarly, the stego errors at stage two are noted just below the third horizontal line. The shaded area states the range of embeddable errors in each round. A pair represented by two down arrowed lines depicts the possible modification of each error in a cycle. The shaded lines are used to highlight the errors, which are crossing the embedding range after the embedding cycle. After the first round, the embeddable errors of stage zero, i.e., -4 to 3, will be dispersed within -8 to 7 in the errors of stage one. The scheme will consider the -4 to 3 valued errors of stage one as embeddable errors for the second round. After executing the embedding process for k-times, the stego pixels $S_{i,j}$ of the corresponding cover pixels, $I_{i,j}$, is formed by adding these modified errors $\tilde{e}_{i,j}$ with its respective predicted values.

5.3.3 Defining the Principalities of Data Extraction and Cover Image Reconstruction in MLMC Schemes

The stego error $\tilde{e}_{i,j}$ is a modified value of $e_{i,j}$ obtained after embedding the message bits into them. While measuring $e_{i,j}$, the sender side applies prediction rules associating a set of pixels. The receiver end generates the same predicted values by the affairs that the context pixels participated in the prediction rules in the encoder side as they were remained unchanged in the stego image by the sender [32] or in the meantime, the same pixels are generated by the decoder [31, 107]. The decoder easily computes stego errors $\tilde{e}_{i,j}$ by subtracting the predicted values from the respective stego contents. The Eq. (5.1) states that in each embedding cycle, the encoder at the sender side modified and dispersed positive valued embeddable errors from 0 to 2*nPEP*-1 and negative valued embeddable errors from -1 to -2*nNEP* while implanting data bits. Consequently, the Eq. (5.2) is applied to extract the message bits from these $[-2nNEP,\ 2nPEP-1]$ ranged stego errors, i.e., from $\tilde{e}_{i,j}$ where $-2nNEP \le \tilde{e}_{i,j} < 2nPEP$. All the modified errors of a cycle are reconstructed by Eq. (5.3).

$$m = \begin{cases} 1 & if \ \ \mathrm{mod}(\left|\tilde{e}_{i,j}\right|, 2) = 1 \\ 0 & Otherwise \end{cases} \tag{5.2}$$

$$e_{i,j} = \begin{cases} \left|\tilde{e}_{i,j} / 2\right| & if \ \ -2nNEP \le \tilde{e}_{i,j} \le -1 \\ \left\lfloor \tilde{e}_{i,j} / 2 \right\rfloor & if \ \ 0 \le \tilde{e}_{i,j} < 2nPEP \\ \tilde{e}_{i,j} + nNEP & if \ \ \tilde{e}_{i,j} < -2nNEP \\ \tilde{e}_{i,j} - nPEP & if \ \ \tilde{e}_{i,j} \ge 2nPEP \end{cases} \tag{5.3}$$

The steps 1-6 of the algorithm 5.2 of Figure 5.7 are executed to extract the whole message stream from the stego prediction errors $\tilde{e}_{i,j}$ and to reconstruct the original prediction errors $e_{i,j}$. Step 1 directs the scheme to start the extraction and reconstruction processes from the *k*-th cycle modified errors. Step 2 extracts the secret message bits by using the Eq. (5.2). This step extracts the message bits that were implanted in this execution cycle by the encoder. Step 3 reconstructs the errors which are primitives regarding this execution cycle, i.e., the errors of stage *k*-1. Steps 4-5 control the execution of the steps 2-3 for *k*-times. Finally, step 6 announces the completion of the task.

As a final point, the decoder reconstructs the cover contents by adding the predicted values with the corresponding $e_{i,j}$. The applied predictor, its parameters, the value of *k*, *nNEP*

and *nPEP* act in a group as a secret key for providing stronger data security at its bit extraction and error reconstruction phase.

Algorithm 5.2*: ExtractionReconstruction(k,* $\tilde{e}_{i,j}$ *)*

Step 1. *Set c=k, where c is used to track the execution cycle.*

Step 2. *Apply Eq.* (5.2) *to extract each of the message bit m of M which were embedded at* c^{th}*-cycle.*

Step 3. *Apply Eq.* (5.3) *to reconstruct the errors which were found after* $(c\text{-}1)^{th}$ *embedding cycle.*

Step 4. *Set* $c = c - 1$.

Step 5. *If c>0, set* $\tilde{e}_{i,j} = e_{i,j}$ *and go to Step* 2.

Else Go to Step 6.

End if

Step 6. *Announce the completion and stop the task.*

Figure 5.7: Secret extraction and cover errors reconstruction processes

5.3.4 Selecting Embeddable Errors for Fixed Payload

In most applications, the length of embedded data, i.e. $|M|$, is a fixed value. Let, the length of the message be *C*. The proposed scheme embeds the target payload of length *C* into the errors of values ranging from -1 to *-nNEP* and from 0 to *(nPEP-1)* by its *k*-cycle data embedment process. In each cycle, 50% of the samples (i.e., distinct valued errors) of the original embeddable errors exceeds the embedding range by the embedding rules, as it is observed in Figure 5.6. This means that the *-nNEP* to *-nNEP/2* valued negative errors and *nPEP/2* to *nPEP-1* valued positive errors, observed in the starting state of the c^{th} cycle, become non-embeddable for the next $(c+1)^{th}$ cycle. This implies that -1 to *-nNEP/2* negative valued and 0 to *nPEP/2* positive valued primitive error points will be reused to embed more data in the $(c+1)^{th}$ cycle. As it is shown in the Figure 5.6, the samples of the embeddable errors are {-4, -3, -2, -1, 0, 1, 2, 3}. All these eight sample errors accept the message bit during their first cycle data embedment phase. After that embedding cycle, the frequencies of errors of -4, -3, 2 and 3 will be redistributed to {-8, -7}, {-6, -5}, {4, 5} and {6, 7} respectively and thus, these errors will exceed the embedding range. Simultaneously, the errors with values of -2, -1, 0 and 1 will be redistributed within the range -4 to 3. In consequence, only the primitive errors with values of -2, -1, 0 and 1 are reusable for bit implantation in the second cycle. As a sample, these 4 errors

{-2, -1, 0, 1} are the 50% of the original samples {-4, -3, -2, -1, 0, 1, 2, 3}. Similarly, the original sample errors -2 and 1 will cross the embedding range by the second cycle as these were dispersed to {-4, -3} and {2, 3} respectively by the first cycle. Consequently, original errors of values of -1 and 0 will remain as reusable for third cycle data implantation. These two samples -1 and 0 are the 1/4 of the original samples {-4, -3, -2, -1, 0, 1, 2, 3}. Now, concentrating to only positive valued errors, this is realized that (0 to $nPEP/2^0$), (0 to $nPEP/2^1$), (0 to $nPEP/2^2$), (0 to $nPEP/2^3$), ..., (0 to $nPEP/2^{k-1}$) valued original errors are reusable as embeddable errors for the embedding cycle of c=1, c=2, c=3, c=4, ..., c=k respectively. A similar pattern of reusability of original errors exists for the negative valued embeddable errors. As a result, after the completion of the k-th cycle, the embedding capacity will be $\sum_{j=0}^{k-1}\sum_{i=-1}^{nNEP/2^j} h(i)+\sum_{j=0}^{k-1}\sum_{i=0}^{nPEP/2^j} h(i)$, where $h(i)$ represents the frequency of an error i, the inner summation computes the payload for an embedding cycle j and the outer summation is used to repeat the inner summation for k times. To meet the demand for the embedding payload, this theoretical capacity should be equal to or greater than C. Hence,

$$\sum_{j=0}^{k-1}\sum_{i=-1}^{-nNEP/2^j} h(i)+\sum_{j=0}^{k-1}\sum_{i=0}^{-nPEP/2^j} h(i)\geq C \tag{5.4}$$

It is notable that different sets of k, $nNEP$ and $nPEP$ values will satisfy the relation '$\geq$' in the Eq. (5.4). Among these set of verified values (which satisfy the relation $\geq$), a tuple is defined as an optimal value for k, $nNEP$ and $nPEP$, for which the left side of Eq. (5.4) generates the minimum value. These optimal values can also be defined by relating these parameters with the embedding layer L of single cycle data embedment process while L layers are just enough to accept C-bits of data in MLSC process. Among values of k, $nNEP$ and $nPEP$ satisfying the Eq. (5.4), one set of values is defined as the optimal one by the Eq. (5.5). Thus, the values of $nNEP$ and $nPEP$ are definable by the Eq. (5.4) and the Eq. (5.5).

$$\underset{k\in\{1,2,...\},nNEP\in\{1,2,...\},nPEP\in\{1,2,...\}}{\arg\min} \quad k(nNEP+nPEP)\leq L \tag{5.5}$$

5.4 Result Analysis and Discussions

The prime objective of the proposed MLMC scheme is to improve the embedding capacity. To evaluate the performance of the proposed scheme, binary data are embedded into 50 Standard images, 500 images of BOSS database and 5000 images of the CalTech image dataset (in total

5550 images) and results are investigated in the scenarios of MLSC and MLMC embedding circumstances. The performance of the MLSC schemes and the proposed MLMC scheme are evaluated on a set of embedding parameters, e.g. payload or capacity, PSNR, structural similarity index (SSIM) value and time complexity. The schemes of Hong in 2012 [31], Leung *et al.* in 2013 [51], Tai *et al.* in 2009 [83], Wang *et al.* in 2014 [94] and Pan *et. al.* in 2015 [114] are experimented as MLSC processes and a comparison between these MLSC schemes and the proposed MLMC scheme are demonstrated in the following discussions. The discussion is provided for the performance parameters listed above. First, in the MLMC scheme, results of multilayer double cycles (MLDC), multilayer triple cycles (MLTC), multilayer quadruple cycles (MLQC), multilayer pentadruple cycles (MLPC), multilayer sextuple cycles (MLSEXC), multilayer septuple cycles (MLSEPC) and multilayer octuple cycles (MLOC) are evaluated among themselves. Thereafter, these are compared with MLSC schemes regarding their performance parameters.

5.4.1 Experimental Setup

Digital data of type text, numerical records, small images having a size of 80x80 pixels, tiny audio data of “amr” format are concealed into an image of size 510x510 in both individually and hybridized manners. Firstly, 500 images are randomly copied from all the 5550 stated images to test the ability of implanting hybrid data of large volume. These images are resized to 510x510 pixels and experimented on a laptop. Thereafter, 5550 images are resized to 210x210 pixels to embed an arbitrary bit stream for the purpose of analyzing and comparing its performance parameters with its competing schemes. The hybrid data is generated from an audio file of ‘amr’ format of 16 seconds duration (reading as characters from the file), the text of 1000 characters and an image of size 80x80. The binary stream M of hybrid data, that is to be implanted, consists of 260720 bits in total. Again, after the message extraction by the decoder, the binary stream of the audio is converted to character streams.

Again, to embed massive data, a random bit generator is used to generate a long binary bit stream comprising of ‘0’ and ‘1’. The MLSC and MLMC schemes are experimented on 5550 images and analyzed thereby sequentially. The data embedment is performed for 1 to 8 cycles in each of the 99 error points separately in the multi-cycle schemes. All the results are outlined in the following subsections.

5.4.2 Analysis of Embedding Payload

In this subsection, the data embedment capability of the schemes in terms of either payloads or embedding capacity is analyzed and compared from different perspectives. First, a mathematical analysis on embedding payload is provided. The requirement of large embedding cycle to achieve massive data embedment is analyzed, thereafter. Next, the average embedding payloads and capacities are investigated under the affairs of fixed embedding layers, fixed embedding cycle, an equal number of encountered errors, sequentially.

5.4.2.1 Mathematical Analysis of the Embedding Payload

Eq. (5.1) states that *-nNEP,* … …, -2, -1, 0, 1, 2, … …, *nPEP-1* valued embeddable prediction errors are modified to {-2*nNEP,* -2*nNEP+1}*,… …, {-4, -3}, {-2, -1}, {0, 1}, {2, 3}, {4, 5}, … … {2*nPEP-2, 2nPEP-1*} respectively in each embedding cycle depending on message bit *m* embedded. Let the frequencies of the original prediction errors of z_- , … …, -1, 0, 1, 2, … … z_+ are $h(z_-)$, … …, $h(-1)$, $h(0)$, $h(1)$, $h(2)$, … … $h(z_+)$ respectively in the prediction error histogram where z_- and z_+ denote the highest valued negative and the highest positve valued error correspondingly. Frequencies of every embeddable prediction error are diffused into 2^c modified errors in the *MLMC* scheme after the completion of *c*-th embedding cycle. For example, the scheme diffuses the errors of 0 and 1 into [0, 3] and [4, 7] ranged modified errors, respectively, after the completion of 2nd cycles, as shown in Figure 5.6. In the first cycle of *MLMC* scheme, all the embeddable errors of *nNEP*, ..., -1, 0, 1, ..., *nPEP* contribute to the payload by an amount of *h(-nNEP)*+... +*h(-1)*+ *h(0)*+ *h(1)*+ ... + *h(nPEP-1*). By this time, 50% of the samples of primitive embeddable errors (SPEE), i.e. 50% of the members of the error set {*nNEP, nNEP+1, ..., ..., 0, 1, ..., ..., +nPEP-1*}, will move to non-embedding range. This means that -nNEP to (-nNEP/2)-1 and nPEP/2 to nPEP valued original errors will cross the embedding range of [*nNEP*, *nPEP*] after the first cycle data embedment task. In the second cycle, only the [-nNEP/2, nPEP/2-1] ranged errors of the SPEE will accept message bits. Consequently, the embedding payload in the second cycle is *h(-nNEP/2)*+... +*h(-1)*+ *h(0)*+ *h(1)*+ ... + *h((nPEP/2)-1)*). Similarly, the third cycle augments the payload by *h(-nNEP/*2^2*)*+... +*h(-1)*+ *h(0)*+ *h(1)*+ ... + *h((nPEP/*2^2)-1). Thus, the total embedding payload for MLMC scheme is approximated by Eq. (5.6).

$$Payload_{MLMC} = \sum_{j=0}^{k-1} \sum_{i=-1}^{-\lfloor nNEP/2^j \rfloor} h(i) + \sum_{j=0}^{k-1} \sum_{i=0}^{\lfloor nPEP/2^j \rfloor - 1} h(i) \quad (5.6)$$

In this equation, the outer summation is used to compute the total payloads that are achieved from *k* cycles. The inner summation repeats for each of the *k* cycles and computes the frequencies of all the embeddable errors in a cycle. The first and second parts of the equation in the right side enumerate the payloads for negative and positive valued errors, respectively.

5.4.2.2 Higher Embedding Cycle for Embedding Hybrid and Massive Data

The average and the maximum payloads in 500 images are measured in MLMC scheme for each of the cycles and layers. These are summarized in Table 5.1. Only the black cell values in the table can meet the demand of a big payload of 260720 bits. The tabulated results state that the single cycle, even the double cycle cannot meet the requirement of that higher embedding capacity within the embedding layer 10. The average payload that is achieved within the embedding layer 8 in the triple cycle and the layer 1 in the quadruple cycle are still lower than the demanded payload. However, all the maximum payloads in the triple cycle are greater than the stated demanded figure.

To investigate in the tabulated data, the matter of attaining higher embedding payloads by the MLMC scheme, let $PL_{l,c}$ represents the payload that is achieved by embedding into *l* layers for *c* cycles. If $l_1 \times c_1 = l_2 \times c_2$ for two different values of *l* and *c*, where $c_2 > c_1$, according to the philosophy of this research the achieved payload in c_2 cycles should be higher than the attained payload in c_1 cycles, i.e., $PL_{l_2,c_2} > PL_{l_1,c_1}$. This behavior is investigated in the experiment and tabulated in Table 5.1. From the maximum payloads of the Table 5.1, it is observed that:

$PL_{1,3}$(=263000bits)>>$PL_{3,1}$(=101000bits), $PL_{2,3}$(=287000bits)>>$PL_{3,2}$(=202000bits), $PL_{2,3}$(=287000bits)>>$PL_{6,1}$(=102000bits), $PL_{2,4}$(=341000bits)>>$PL_{4,2}$(=202000bits) and $PL_{2,4}$(=341000bits)>>$PL_{8,1}$(=102000bits), where >> stands for much greater than.

These analyses establish that payloads obtained by employing higher cycles are larger.

Table 5.1: Achieved the average (Avg.) and the maximum (Max) payloads (x1000) among the 500 images of size 510x510 at various embedding layers and embedding cycles.

Cycle→	1		2		3		4	
Layer↓	Avg.	Max	Avg.	Max	Avg.	Max	Avg.	Max
1	79	100	154	199	215	263	254	303
2	82	101	159	201	231	287	276	341
3	85	101	164	202	237	302	293	364
4	87	101	167	202	247	303	304	380
5	88	102	170	203	248	303	315	390
6	89	102	173	203	250	304	322	399
7	90	102	175	203	254	304	327	404
8	91	102	177	204	257	304	331	404
9	92	103	179	204	261	305	334	405
10	93	103	180	205	262	305	337	405

This is also investigated that, to achieve the same capacity of k cycle embedding into L/k layers, the multilayer single cycle scheme has to embed more than L layers, i.e., when $PL_{l_1,c_1} \approx PL_{l_2,c_2}$ for $c_1 = 1$, $c_2 > 1$ then $l_1 \times c_1 > l_2 \times c_2$ where $\approx$ stands for close to. The scenario is already observed in Figure 5.2, where MLSC scheme utilizes 9 peaked errors of the error histogram, i.e. layer 4, to embed 25070 bits whereas the MLDC scheme uses 3 errors for layer 1 double time embedment to obtain the same payloads. Table 5.1 also justifies the claim. The figures in the table state that, $PL_{4,1}$(=101000bits)<<$PL_{1,2}$(=199000bits) though $l_1(=4) >> l_2(=1)$. As always $l_1 \times c_1 > l_2 \times c_2$ and in some instances $l_1 \times c_1 >> l_2 \times c_2$ the non-embeddable errors in the single cycle scheme have to be shifted by more amount, i.e., non-embeddable negative errors by $-l_1$ and positive errors by l_1 in the MLSC scheme whereas negative and positive valued non-embeddable errors by $-l_2 \times c_2$ and $l_2 \times c_2$ respectively in the MLMC scheme. During these shifting of pixel values, from 0 to l_1 valued and 255-l_1 to 255 valued pixels in MLSC scheme and from 0 to $l_2 \times c_2$ valued and 255-$l_2 \times c_2$ to 255 valued pixels in MLMC scheme exceeds the gray range. The number of grayscale exceeding pixel values is higher in the MLSC scheme. This postulate implies that the MLSC schemes have to record positions for more pixels in a location map [51] to overcome the underflow and overflow problem in the grayscale than that is for the MLMC scheme. The pixels in the

location map are not used to embed data in most of the instances. This means that the quantity of message implantable pixels will be reduced in the MLSC schemes.

The management policy of that location map is not discussed in this scheme because this research has not contributed to any aspect of the location map. Rather, it is assumed that the location map will be handled as it is done by the other schemes. Nevertheless, the experiment reveals that it is possible to avoid the management of location map if the scheme is free from any restriction of choosing the cover image. The investigation ensures that 20%, 0%, 8%, 42%, 30% and 6% images of the dataset of Natural, BOSS, CalTech, Standard, Texture and Satellite, respectively, do not contain any pixel which is smaller than 6 or greater than 249. Hence, we have many images to implement double cycle embedment in layer 3 and triple cycle in layer 2 without maintaining any location map. It is also found that, there are a good number of images in these datasets where higher cycles and layers are implementable without handling the overflow and underflow issues. In that case, the scheme has to select a cover image only from these images, which will not introduce any underflow or overflow problem in the stego image.

From the discussion, it is understood that to meet the larger embedding payload, higher cycles should be employed. Again, to verify the reconstructed audio quality, in the experiment, the extracted audio data are first separated from the hybrid data. Each of the eight bits of the audio data is written as a character in a file and the file is named with “amr” extension. Though, some additional noises are introduced into the retrieved audio stream, the audio information is fully realizable. Neither the embedding rules nor the extraction process is responsible for the introduction of these noises. This happens for reading the audio data as characters by the data hider from the audio file of “amr” format and saving the retrieved data as a character by the decoder. Nevertheless, the proposed scheme leads all other existing schemes in hiding small audio data, e.g., small audio of instructions, statements, investigations or interviews, in a hybridized manner.

5.4.2.3 Analyzing Average Embedding Payload and Embedding Capacity in MLMC Scheme

In the MLMC schemes, data are implanted into a wider range of error points depending on the requirement of the embedding payloads. In our experiment, the embedding is done in 2 to 99 error points separately, i.e. up to 49 layers, to analyze the behavior of our scheme at higher layers. The average payloads of n images for each nEP embeddable error points are computed by the Eq. (5.7) for $2 \leq nEP \leq 99$.

$$AvgPayload(nEP) = \frac{\sum_{r=1}^{n} payload(r, nEP)}{n} \tag{5.7}$$

where $payload(r, nEP)$ returns the value of embedding payload that is obtained by implanting bits into *nEP* error points in the *r-th* image. All average embedding capacities depicted in this chapter are measured by *AvgPayload(nEP)/(*image_size).

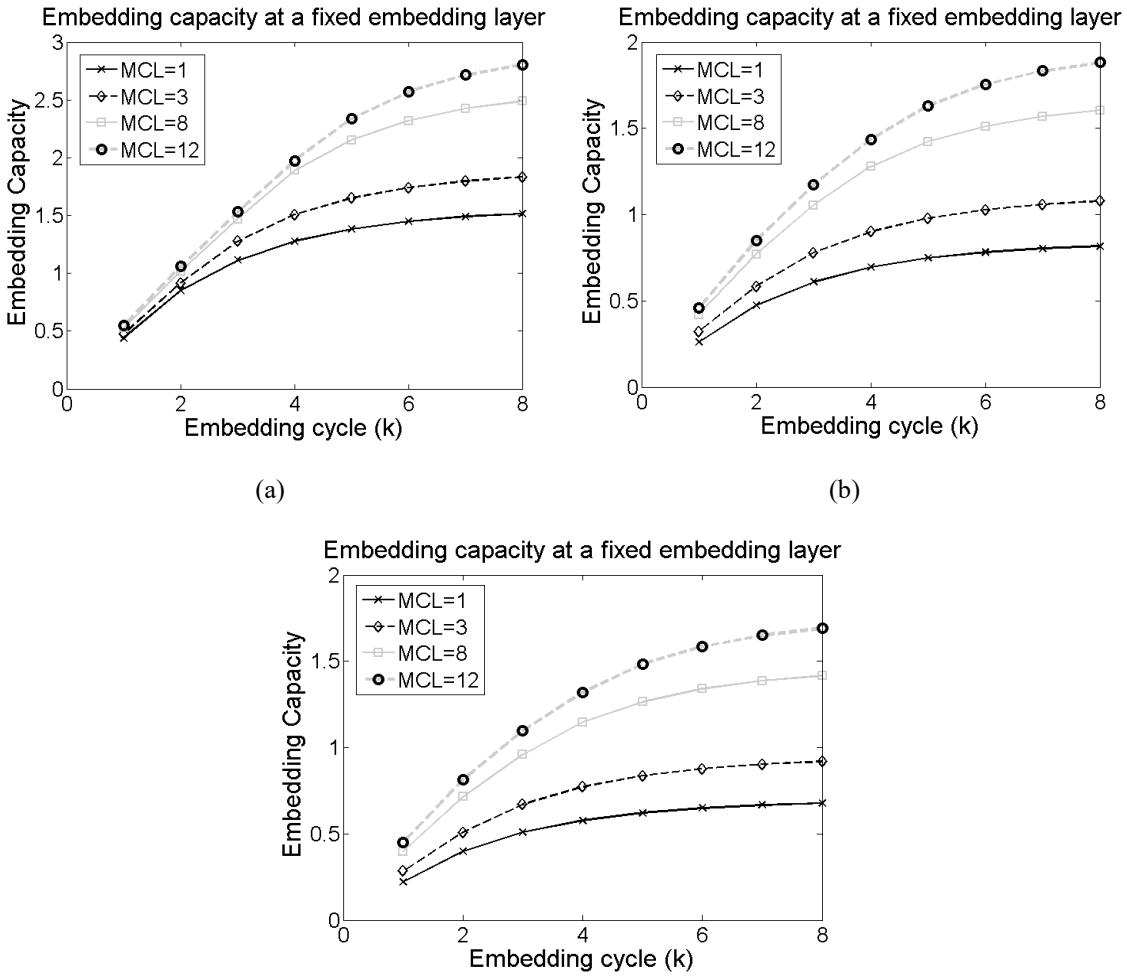


Figure 5.8: Average embedding capacity achieved in each of the embedding layers 1, 3, 8 and 12 in (a) BOSS images, (b) CalTech images and (c) Standard images. The capacities are investigated at different embedding cycles.

5.4.2.3.1 Investigating Average Embedding Capacity under the Fixed Embedding Layer

In this experiment, the behaviors of embedding capacity are investigated in each individual embedding layer for embedding cycle ranges from 1 to 8. The average capacities achieved in layer 1, 3, 8 and 12 are drawn against 8 embedding cycles in Figure 5.8 (other layers are avoided for clear depictions) separately for BOSS, CalTech and Standard image datasets in the Figure 5.8(a), Figure 5.8(b) and Figure 5.8(c) respectively. The demonstrated results for each of the embedding layers 1, 3, 8 and 12 state that the embedding capacity increases with the increment of embedding cycles. It is also noticeable that the capacity enhances in the upper embedding layers. These similarities in the behaviors of embedding capacities are observed in all the image datasets. If the higher layers are employed, e.g. EL=8 or EL=12, then within embedding cycle 4, the capacity is improved by a factor of about 2 with respect to its immediate lower cycle. In overall, geometrical progressions in the embedding capacities are investigated with the advancement in the embedding cycles.

5.4.2.3.2 Investigating Embedding Capacity under the Fixed Embedding Cycle

In this experiment, each time the embedding cycle c is defined to a fixed value, say 1 for the first time. The embedding is performed in each of the 99 embeddable errors for each embedding cycle separately. The embedding capacities of some sample embedding cycles 1, 3, 5 and 8 for first 25 embeddable errors (for better depiction) are demonstrated in the Figure 5.9(a), Figure 5.9(b) and Figure 5.9 (c) independently for the same three image datasets. These figures state that the embedding capacity increases both with the employment of higher embedding cycles and more embeddable error points. Though the embedding capacity increases sharply for first 5 embeddable errors, thereafter, the rate of increment in the embedding capacity decreases gradually because the accounted higher valued errors (resides in the upper layer) contribute less to the payload as the frequencies of these errors are smaller. Nevertheless, the achieved capacities in the higher cycles are much greater than the capacities achieved in MLSC schemes (when k=1). Another perceptible issue is that the embedding capacities in 3 cycle process are more than the double of the single cycle process. This rate of boosting up the embedding capacities by the process of 5 and 8 cycles reduces gradually. The reason lies in the shifting properties of the embeddable errors. It is already discussed that after each completed cycle, 50% of the members of SPEE, i.e. 0.5($nNEP$+$nPEP$), move to the non-embedding range. Consequently, a good number of embeddable errors become non-embeddable after the first few cycles due to their movement by the embedding rules. It implies

that the frequencies of the embeddable errors decrease dramatically after a few of the cycles. Consequently, the rate of contribution to the embedding payload declines in each of its next cycles. Though the rate of capacity improvement decreases in the upper cycles, the total capacity of each higher cycle is greater than the single cycle.

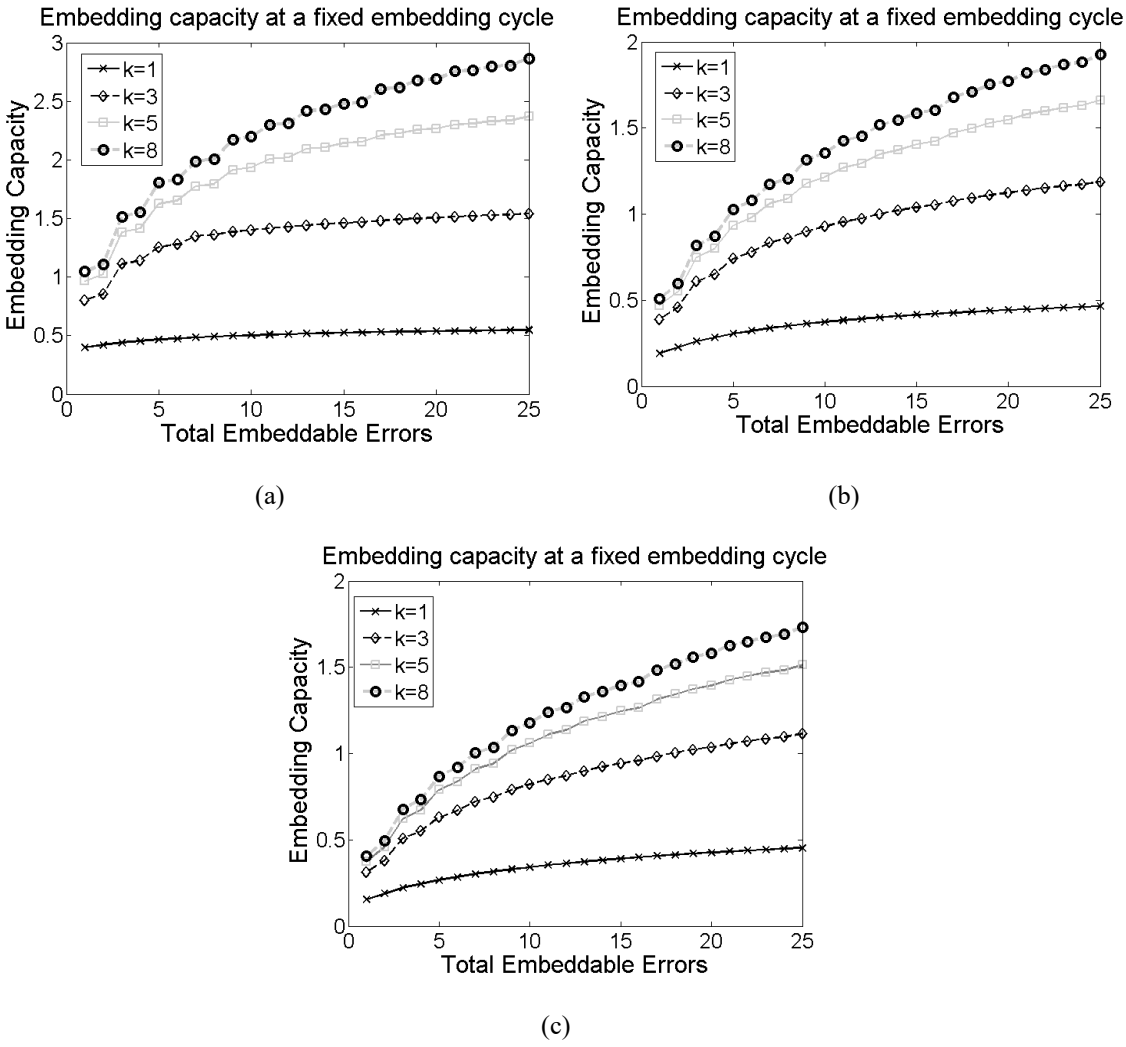


Figure 5.9: Average embedding capacity in each of the sample cycles 1, 3, 5 and 8. The results are investigated for embeddable errors from 1 to 25 in image dataset of (a) BOSS (b) CalTech and (c) Standard.

5.4.2.3.3 Investigating Embedding Payload at Equal Number of Encountered Error Points

Figure 5.10 demonstrates the achieved average embedding payloads in the different embedding cycles. The number, annotated at the top of each bar, indicates the quantity of embeddable error points nEP_c (=$nNEP$+$nPEP$) that are applied in the cycle c. The values of nEP_c in each cycle c are selected in such a way that it becomes equal to or close to nEP_1/c, i.e. $nEP_c \approx nEP_1/c$. In the Figure 5.10, nEP_1 is 24. Therefore, total encountered error points are 24, 24, 24, 24, 25, 24, 28 and 24 in the cycle 1, 2, 3, 4, 5, 6, 7, and 8 respectively. As the values of c and nEP_c are always integer, for c=5 and c=7, total encountered error points, which are 25 and 28 respectively, are more than 24; because these are no alternative values which are closer to 24. This way the encountered error points in all the cycles are equalized at the time of comparing the average payloads in the different cycles individually in each image datasets as depicted in Figure 5.10(a), Figure 5.10(b) and Figure 5.10(c). The payload for a particular cycle increases with the increment in the embedding cycles from 1 to 5. Thereafter, it decreases gradually for each cycle. Though, the 7-cycle embedding process demonstrates a bit improvement in the embedding payload compared with 6-cycle embedding process, it considers nEP_7= 4, i.e. total 4×7=28 encountered error points, instead of nEP_7=24/7 due to the fractional value in 24/7. Hence, a smaller increment in the embedding payload is observed; indeed, the payload decreases there too. The decrements in the payloads at the higher embedding cycles, e.g. c>5 in the Figure 5.10, are very rational because, thereafter, the quantity of nEP_c decreases multiplicatively (i.e. by c as nEP_c=nEP_1 /c). Besides, after each embedding cycle, half of the primitives in OSEPE are shifted to the non-embeddable area.

It is already observed in Figure 5.2 that in the PEH $h_i >> h_{i+1}$ and $h_{-i} >> h_{-i-1}$ for smaller i. Hence, for smaller values of nEP_c, the shifting of 50% of nEP_c to a non-embedding range implies that a larger quantity of embeddable error points will cross the embedding range by the embedding rules after the completion of each cycle. Consequently, after executing the embedding process for a few of cycles into nEP_c embeddable errors, the added payloads by these embeddable errors with remaining frequencies become nominal. However, the achieved payloads in cycle 6, 7 and 8, (depicted in the last three bars in Figure 5.10), are much greater than the achieved payload by equal encountered error points in MLSC scheme, (depicted in the first bar in the same figure). This implies that a multi-cycle process is better capacity generative than single cycle processes.

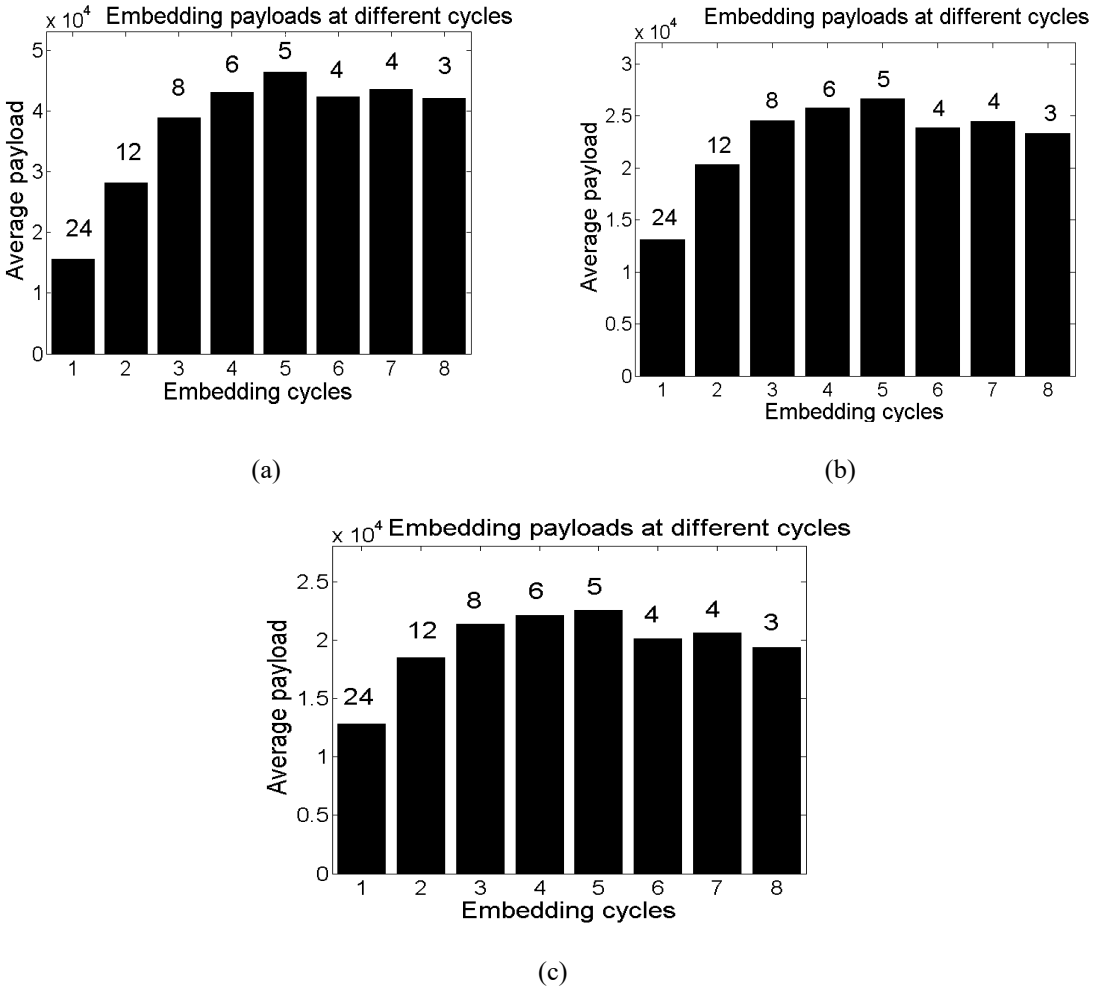


Figure 5.10: The average payloads in various embedding cycles *c*; in equal encountered error points; and in the image dataset of (a) BOSS, (b) CalTech and (c) Standard. The value on the top of each bar indicates the nuuummmb of embeddable errors, *nE*, in the cycle. Total encountered errors in each cycle *c* are $\boldsymbol{nE} \times \boldsymbol{c}$, e.g., these are 24, 24, 24, 24, 25, 24, 28 and 24 in cycle 1, 2, 3, 4, 5, 6, 7, and 8 respectively.

5.4.3 Analysis of PSNR Value

Mean square error (MSE) is used to measure the value of PSNR. Smaller values of MSE means higher value of PSNR. PSNR is used to measure the level of visual quality of an image compared with the original. The larger value of PSNR implies as the better image quality. An embedding scheme, therefore, considers having larger PSNR value.

5.4.3.1 Mathematical Representations of MSE in MLMC Scheme

From Figure 5.6, it is observed that after the completion of *c* embedding cycles, the '0' valued errors are modified by $x_j,\ 0 \le j < 2^c$ while $x_j \in [0, 2^c - 1]$. The embedding rules apportion all the positive valued embeddable errors *i*, $0 \le i \le nPEP$, to the range $[\ 2^c \times i,\ 2^c \times (i+1) - 1]$, e.g. '1' diffuses from 8 to 15 for 3 cycles data embedment when $nPEP >= 8$. The total shifting amount of the embeddable errors depends on the quantity of bit '1' in the message stream, embedding layers and cycles. Each positive valued non-embeddable error is shifted by $nPEP \times 2^{k-1}$ after the completion of *k* cycles. For the simplicity, consider the frequency of each error *i*, i.e. h_i, is equally distributed into $[\ 2^c \times i,\ 2^c \times (i+1) - 1]$ ranged 2^k modified errors after the completion of *k* embedding cycles. The issue of crossing the embedding range by the embeddable errors is also ignored here for the minimalism. These two considerations are taken into account just to measure approximate distortions. Then, a summation of the square of the stego displacement (SSSD) for the positive valued errors is measured using the Eq. (5.8).

$$SSSD_{MLMC+} = \sum_{i=0}^{nPEP-1} \sum_{j=0}^{2^k-1} \left(j + i \times 2^k \right)^2 h_i / 2^k + \sum_{l=nPEP}^{z_+} \left(nPEP \times 2^{k-1} \right)^2 h_i \tag{5.8}$$

The inner summation $\sum_{j=0}^{2^k-1}$ sums up for the distortions that are introduced by the frequencies of a single error *i*, i.e. h_i, due to its diffusions into $[\ 2^k \times i,\ 2^k \times (i+1) - 1]$ modified errors. The outer summation $\sum_{i=0}^{nPEP-1}$ repeats the inner summation for each of the embeddable errors from 0 to nPER-1. The third summation $\sum_{l=NPEP}^{Z_+}$ computes the distortions occurred by the non-embeddable errors. Similarly, the SSSD for the negative valued errors in MLMC scheme, $SSSD_{MLMC-}$, is defined by the Eq. (5.9). Finally, the total *MSE* is calculated using the Eq. (5.10).

$$SSSD_{MLMC-} = \sum_{i=0}^{-nNEP} \sum_{j=0}^{2^k-1} \left(-j + (i+1) \times 2^k \right)^2 h_i / 2^k + \sum_{l=nNEP-1}^{z_-} \left(nNEP \times 2^{k-1} \right)^2 h_i \tag{5.9}$$

$$MSE_{MLMC} = \frac{SSSD_{MLMC+} + SSSD_{MLMC-}}{h \times w} \tag{5.10}$$

where *h* and *w* stand for the height and width of the image.

5.4.3.2 Analysis of the Ratio of Payload Per PSNR in the Schemes

Analyzing on payload per PSNR, it is investigated that the MLMC scheme can manage a better trade-off between the PSNR, embedding payloads and embedding cycles at or above the PSNR of 30dBm (lower margin of visually distortion realization [113]). The average payload per PSNR for each embedding cycle is delineated in Figure 5.11.

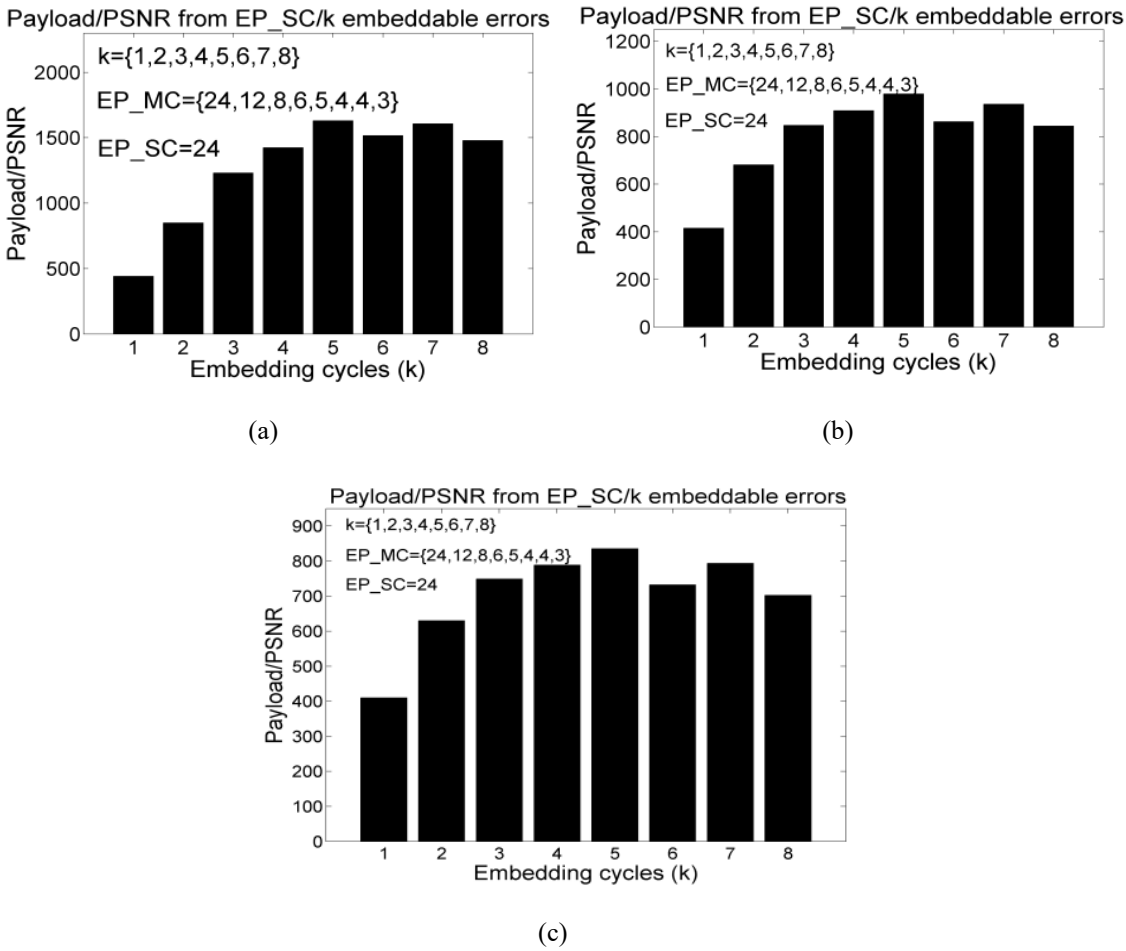


Figure 5.11: The amount of payload per PSNR against each embedding-cycle when the number of encountered error points are about equal. The number of embeddable error points in multilayer, single cycle is *EP_SC*. Here *EP_SC*=24. The number of embeddable error points in multi-layer multi-cycle is *EP_MC*=ceil (*EP_SC/k*). Here, the values of *EP_MC* are {24,12,8,6,5,4,4,3} for cycles {1,2,3,4,5,6,7,8} respectively. The experimental values for *EP_SC* and *EP_MC* are annotated in each figure (a) -(c) for BOSS, CalTech and standard image datasets.

The results are depicted separately for BOSS images, CalTech images and Standard images respectively in the Figure 5.11 (a), Figure 5.11(b) and Figure 5.11(c). In all the figures, the ratio between the payload and the PSNR increases for the first five embedding cycles.

Thereafter, it starts to decrease because the image distortions in the higher embedding cycles increases by the multiple of about $2^{(k-1)^2}$, e.g. by 2^{0^2} for first cycle, 2^{1^2} for second cycle, 2^{2^2} for third cycle and so on, which are inferred from Eq. (5.10.1) and Eq. (5.10.2). Besides, the rate of increment in the payloads after 5th cycles decreases, as shown in Figure 5.7. The reasons of decreasing the rate of increment in the embedding payloads are described during the analysis of payloads in this figure. Consequently, the payloads per PSNR decrease after embedding cycle 5. The same results are observed in the images of all the image datasets. However, regarding the single cycle process, depicted in the first bar in the Figure 5.11, the payloads per PSNR in all the higher cycles are still in dominating level.

5.4.3.3 Analysis of the Ratio of PSNR Per Capacity in the Schemes

Figure 5.12 demonstrates the behaviors of PSNR per embedding capacity in all the embedding cycle processes. As the embedding cycle goes higher, the embedding capacity increases. Meanwhile, the PSNR decreases. However, above the embedding capacity of about 0.459bpp, 0.245bpp and 0.21bpp respectively in BOSS images, CalTech images and Standard images as shown in Figure 5.12(a), Figure 5.12(b) and Figure 5.12(c), the multi-cycle embedding processes demonstrate remarkably better PSNR. This implies that when the requirement of embedding capacity is higher, MLMC schemes provide better stego image quality.

5.4.3.4 Investigating PSNR among the MLMC Schemes

It is investigated in the Figure 5.12(b) that after the embedding capacity of 0.245bpp, double-cycle embedding process exhibits improved PSNR value and that superiority of proving higher PSNR by double cycle process continues up to embedding capacity of 0.5bpp. Within that embedding capacity range, PSNR in the double cycle embedding process varies from 47.5dBm to 38.2dBm as shown in Figure 5.12(b). Again, after the embedding capacity of 0.5bpp, triple cycle embedding process takes the leads on providing enhanced PSNR value. That leads continues until the PSNR value of the triple cycle falls below 30.8dBm. At the 30.8dBm, triple cycle embedment process reaches its embedding capacity to 0.855bpp. Though, thereafter, quadruple and its higher cycles provide improved PSNR regarding the first three cycles, the PSNR falls below 30dBm. The PSNR value of less than 30dBm indicates noticeable image distortions [113]. Therefore, in the CalTech images, if the required embedding capacity is 0.246bpp to 0.5bpp, the double cycle embedding process meets the requirement of the

embedding capacity as well as manages the image quality. The triple cycle is recommended only to meet the requirement for larger embedding capacity. The quadruple and its higher cycles demonstrate worsening results and thus these are not recommended in the applications of image quality controlled based reversible steganography schemes.

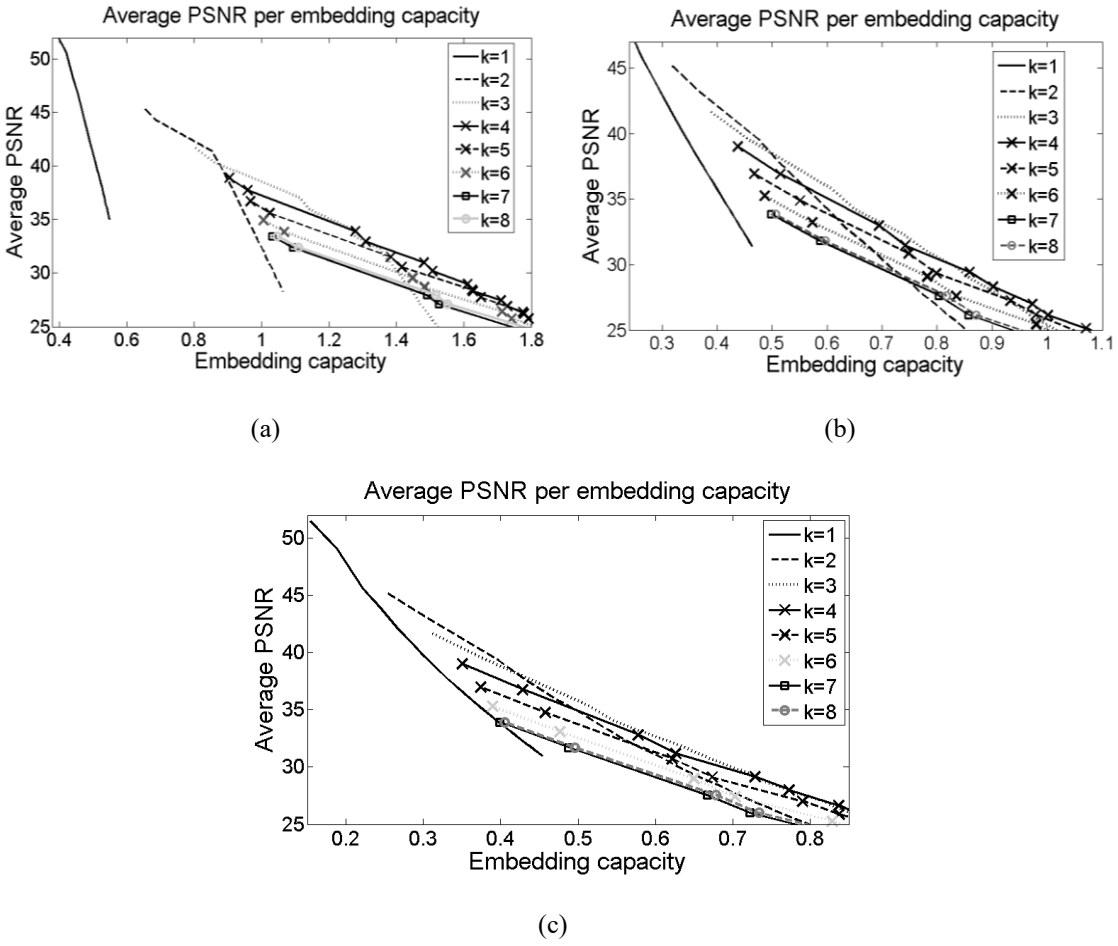


Figure 5.12: Average PSNR per embedding capacity in different embedding cycles in the images of (a) BOSS, (b) CalTech and (c) Standard dataset.

5.4.3.5 Analysis of Embedding Capacity and Embedding Cycles at a Fixed PSNR

The target of the scheme is to meet the demand for higher embedding capacity and in the meantime to manage the PSNR value of 30dBm or more. The reason for analyzing embedding capacity at PSNR level of 30dBm is that the effect of the modification in an image is not visually sensible if PSNR is not less than 30dBm. The Figure 5.12 demonstrates the behaviors of PSNR per embedding capacity in all the embedding cycle processes. The figure states that

the embedding cycles from 1 to 8 provide the embedding capacities of 0.59bpp, 1.05bpp, 1.4bpp, 1.53bpp, 1.49bpp, 1.43bpp, 1.35bpp and 1.36bpp in the BOSS image dataset, 0.485bpp, 0.71bpp, 0.815bpp, 0.82bpp, 0.785bpp, 0.72bpp, 0.685bpp and 0.695bpp respectively in the CalTech image dataset and 0.48bpp, 0.625bpp, 0.7bpp, 0.7bpp, 0.65bpp, 0.61bpp, 0.56bpp and 0.57bpp respectively in the Standard image dataset at the PSNR value of 30dBm. These notated capacities conclude that at the same deterioration level, for PSNR of 30dBm, multi-cycle processes provide higher embedding capacity and the trend of progress in the embedding capacity continues till 4th cycles. Though, thereafter, the embedding capacity decreases, these are still better than MLSC and more even than the first few of the cycles.

From the analysis on the Figure 5.11 and Figure 5.12, it can be concluded that multi-cycle embedding schemes present higher PSNR values during the embedment of large data.

5.4.4 Performance Comparison with Other Schemes

In this subsection, the results of the proposed MLMC scheme are compared with the reviewed MLSC schemes, e.g. the schemes of Hong (2012) [31], Leung *et al*. (2013) [51], Tai *et al*. (2009) [83], Wang *et al*. (2014) [94] and Pan *et al*. (2015) [114]. The image dataset of CalTech is a rich one (consisted of 5000 images) among our experimented datasets. Hence, only the results obtained in that dataset are demonstrated in Figure 5.12, Figure 5.14 and Figure 5.15.

5.4.4.1 Comparing the Payloads among the Schemes as a Measure of Performance

Figure 5.13 demonstrates the achieved payloads in different schemes at their various embedding layers. Among the MLSC schemes, the most worsening results are investigated in the scheme of Pan *et al.* [114] for several reasons. Firstly, it embeds data by modifying pixel histogram rather than PEH. Secondly, though it embeds data into two different pixel values of $B_p - 1$ and B_p+1, it leaves the highest frequency pixel B_p as unchanged. Thirdly, it skips the block when B_p-1 or B_p+1 does not exist in the histogram (when B_p becomes the leftmost or the rightmost bin in the histogram) or the frequency of, at least one, them is zero. The payloads in the schemes proposed by Tai *et al.*[83], Hong [31], Wang *et al.*[94] and Leung *et al.* [51] are very close and mainly differ by a small quantity for their prediction policies and histogram properties. Among these schemes, the scheme of Leung *et al.*[51] provides a bit higher payloads because that scheme estimates pixel values more accurately in a smaller sized image block. Again, among the multi-cycle schemes, each upper cycle takes the lead on its immediate

lower cycle. The rates of payload improvements in the double cycle and the triple cycle processes happen on an extensive scale. The improvement rate is moderate in quadruple and pentadruple cycles compared with their lower cycles. In the upper cycles of these two, the payload increase is slight and gradual. As a whole, the payload increases in the MLMC processes and the payload improvement is noticeable toward their upper cycles. To explain that claim, consider the payload of MLTC in layer 3. Figure 5.13 depicts that this payload is 4.95×10^4bits. To meet the equal number of embeddable error points, layer values in MLQC, MLDC and MLSC will be 2.25, 4.5 and 9 respectively (fractional value of layer is taken only for the comparisons). In these equalized embedding layers, the payloads in MLQC, MLDC, and the schemes proposed by Tai *et al*.[83], Hong [31], Leung *et al.*[51], Wang *et al.*[94] and Pan *et al.* [114] are 5.1×10^4bits, 4.2×10^4bits, 2.1×10^4bits, 2.74×10^4bits, 2.9×10^4bits, 2.8×10^4bits and 1.2×10^4bits respectively. These quoted values state that payloads obtained in MLMC processes are some multiples of the obtained payloads in MLSC schemes. These values also elucidate that higher cycles provide improved payloads.

Another experiment is carried out to test the behaviors of the minimum and the maximum payloads among the experimental results in a specific layer. The results are demonstrated in Figure 5.14. Regarding the maximum payload, the multi-cycle scheme depicts higher values. Even the minimum payload of MLQC and its upper cycles are very close to the maximum payloads of the MLSC schemes and higher than some of the maximum payload in MLSC, e.g., the schemes proposed by Pan *et al*. [114] and Tai *et al*. [83].

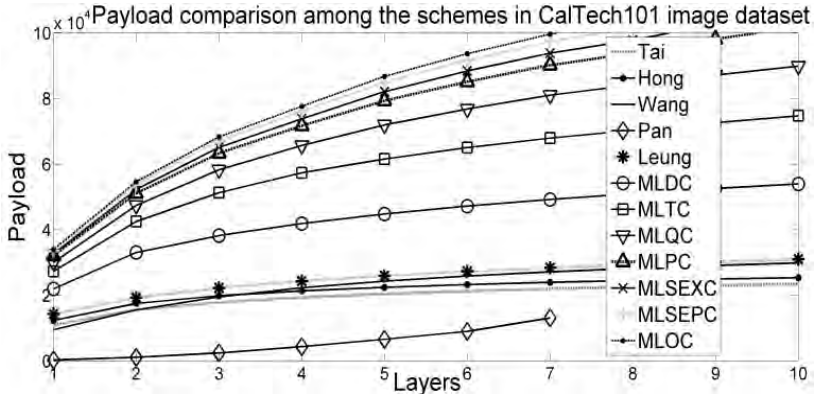


Figure 5.13: Average payload in different schemes.

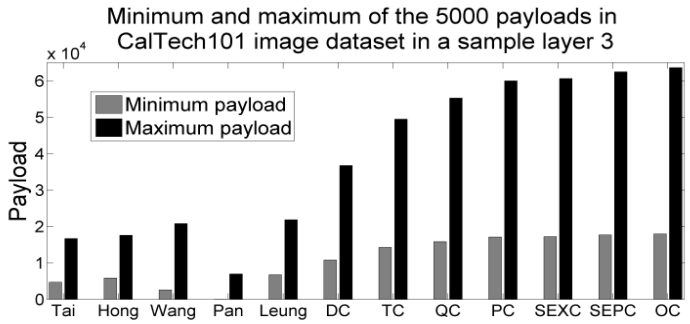


Figure 5.14: The Minimum and the maximum payloads which are obtained from the results of 5000 images of the CalTech image dataset in a sample layer 3 in the different embedding processes.

5.4.4.2 Comparing the Stego Image Quality among the Schemes

In this subsection, the values of PSNR and SSIM are analyzed to measure the quality of stego image. When message bits are implanted into the pixel values of an image, the embedding scheme changes the original statistical cover information in the stego image. PSNR is used to detect the variation of intensities of pixel values in an image regarding another image. Nevertheless, the PSNR does not provide the structural changes in an image. The SSIM is very useful in detecting structural changes in an image. The experiments are conducted on these two parameters and the results are discussed in the following two subsections.

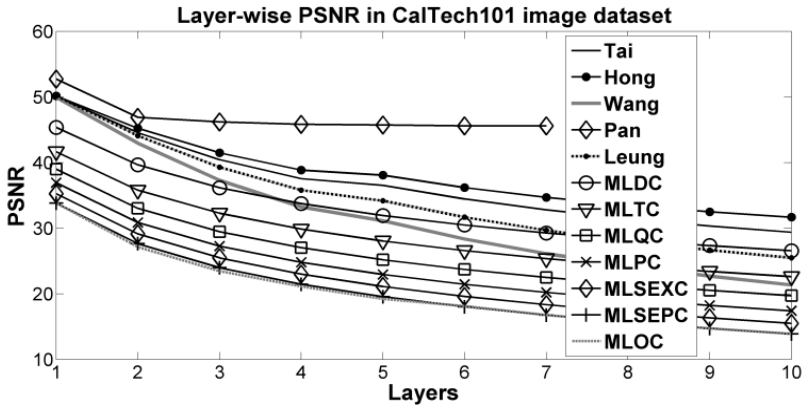


Figure 5.15: The average PSNR in MLSC and MLMC schemes in different embedding layers.

5.4.4.2.1 Comparing the PSNR Values among the Schemes

Figure 5.15 depicts an average of PSNR values (computed from 5000 images) separately in each embedding layer for MLSC and MLMC schemes. It delineates that Pan *et al.*[114] exhibits outstanding performance because the scheme does not change the highest frequency pixels in the block while implanting data. Besides, during implanting into second and higher embedding layers, this scheme reuses the stego image. Before each reuse, this partitions the stego image varying the size of image block and measures the highest frequent pixel in each block to embed data bits into its immediate smaller and greater valued pixels. During each reuse, the new peak takes place at different positions in the histogram of block pixels. Consequently, many pixels, which were shifted in the left direction in their earlier cycles, might be shifted in the right direction by the properties of the histogram in their next cycle. Therefore, the overall displacement of pixel values decreases. The figure also demonstrates that all the MLSC schemes provide larger PSNR values than all the multi-cycle schemes. Among the multi-cycle schemes, the PSNR values decrease as the embedding cycle goes higher. Though, the figure discourages one to choose multi-cycle processes, this does not depict the PSNR at an equal number of encountered error points or the PSNRs per embedded bit. For example, at the embedding layer 2, PSNR in MLDC process is about 40dBm. The PSNR values in the schemes proposed by Tai *et al.*[83], Hong [31], Leung *et al*. [51], Wang *et al.*[94] and Pan *et al.* [114] are about 37.5dBm, 38.5dBm, 35dBm, 33.2dBm and 45.8dBm respectively at equal number of encountering error points, i.e., L=4. Similarly, the PSNR at L=2 in MLTC and at L =6 in the schemes proposed by Tai *et al.*[83], Hong [31], Leung *et al*.[51], Wang *et al.* [94] and Pan *et al.* [114] are about 35.3dBm, 34dBm, 34.9dBm, 31dBm, 29dBm and 45.5dBm respectively. The PSNR in the MLDC and the MLTC schemes are higher than the respective PSNR in the MLSC schemes rather than the scheme proposed by Pan *et al.*[114]. However, this method provides very small embedding capacity. Indeed, when the target is to embed a large amount of data in a single image and the size of data is fixed, the MLMC schemes provide better PSNR value. Single layer schemes like those proposed by Tai *et al*.[83], Hong [31], Leung *et al.*[51], Wang *et al*. [94] and Pan *et al*. [114] can only be used to meet smaller embedding capacity.

5.4.4.2.2 Comparing the SSIM Values among the Schemes

SSIM is widely used to compare the structural similarities between the two images. The results of average SSIM obtained in the CalTech image dataset are tabulated in Table 5.2. The

schemes proposed by Pan *et al.* [114] and Hong [31] yield higher similarity index values between the cover and stego image because these two methods skip lots of pixels without modifying by their block skipping criteria and embedding rules. The proposed scheme provides very competitive SSIM value compared to the schemes proposed by Tai *et al.*[83], Leung *et al.* [51] and Wang *et al.*[94]. MLMC processes are better than all the other MLSC schemes regarding the average SSIM values per embedding bit. For example, average SSIM is 0.854 for L=3 and k=2. Considering an equal number of encountering error points (i.e., at L=6 for MLSC), average SSIM in Wang *et al.* [94] is 0.843. On the other hand, the rate of boosting up the embedding capacity is significantly higher in MLMC scheme, e.g., in Figure 5.13, at L=3, k=2 (MLDC), embedding payload is 4.00×10^4bits and at L=6, payloads in the single cycle schemes of Tai *et al.*[83], Hong.[31], Leung *et al.*[51], Wang *et al.* [94] and Pan *et al.* [114] are 2.0×10^4bits, 2.1×10^4bits, 2.65×10^4bits, 2.5×10^4bits and 1.01×10^4bits respectively. These analyses of results prove that distortion per embedding capacity is smaller in MLMC.

5.4.4.3 Analyzing the Complexity

The time complexity of the proposed scheme is compared experimentally with the different competing schemes by analyzing the required embedment times. For this, at first, the embedding layer is set to 7. Then, each scheme separately embeds same data bits into CalTech images. The total time required o embed data bits into 5000 images is recorded. The average time spent for each image is tabulated in Table 5.3. The scheme proposed by Pan *et al.* [114] presents smallest time complexity because it does not perform any predictions. Rather, it embeds data bits by modifying the pixel histogram. The associated affairs in computing the pixel differences in the scheme proposed by Tai *et al.* [83] and prediction errors in the scheme proposed by Leung *et al*. [51] are less complex than measuring the prediction errors in the scheme proposed by Hong [31]. The required times in these two schemes are smaller than the required times in the scheme proposed by Hong [31]. The scheme proposed by Wang *et al.*[94] takes the highest amount of time to complete the embedding task. Each time it embeds message bits into a single layer and repeats the process for m-times for an m-layer data embedment method. At each embedding layer, it embeds data bits into the highest peaked errors in the error histogram. The embedding rules perform the data embedment by shifting the errors histogram. It employs the same histogram calculation method and an error shifting policy at its consecutive embedding layers. Thus, for m-layer data embedment, it computes the histogram for m-times and applies the error shifting policy for m-times. For that reason, it

takes much time. On the contrary, in each cycle, the proposed scheme accesses and shifts the errors for a single time only during the operation of *m*-layer data embedment. The error shifting policy is repeated for each embedding cycle and not for embedding layers. Therefore, the time complexity increases along with the increment of embedding cycles. Nevertheless, time complexities in MLDC, MLTC and MLQC are very close to that of other MLSC schemes and much smaller than the time required by the scheme proposed by Wang *et al*.[94].

Table 5.2: Comparison of average values of SSIM in the different schemes

Layers (L)	MLDC (k=2)	MLTC (k=3)	MLQC (k=4)	Tai	Hong	Leung	Wang	Pan
1	0.886	0.862	0.839	0.874	0.971	0.895	0.875	0.972
2	0.867	0.831	0.791	0.873	0.970	0.891	0.873	0.971
3	0.854	0.806	0.755	0.872	0.969	0.887	0.868	0.969
4	0.842	0.781	0.717	0.871	0.968	0.884	0.861	0.967
5	0.831	0.762	0.691	0.869	0.967	0.881	0.853	0.964
6	0.819	0.740	0.659	0.868	0.966	0.877	0.843	0.961
7	0.810	0.724	0.638	0.866	0.964	0.874	0.832	0.959

Table 5.3: Execution time per image (in second)

Schemes	Tai	Hong	. Wang	Pan	Leung	DC	TC	QS	PS	SEXC	SEPC	OC
Time (in sec)	3.8	4.5	16.3	3.1	3.9	4.9	5.7	6.4	7.3	9.1	10.9	12.3

5.5 Resistance to Statistical Attacks

There are many steganalysers [112], which can measure the probability of the presence of hidden data inside an image. In the experiments, two latest steganalysers are employed to test the resistance of the proposed scheme against the statistical attacks. As in the upper section, the higher embedding cycles are not recommended for data embedment due to the objective of maintaining the image quality; only first four cycles are analyzed in this section. The descriptions of these steganalysers, their experimenting methodologies, obtained results and the discussions are provided in the following sub-sections.

5.5.1 Security Analysis in SPAM Features

Subtractive pixel adjacency matrix (SPAM) based steganalysis [19, 73] is a very effective and well-known method. The SPAM features are measured by subtracting the adjacent pixels. The differences of adjacent pixels in two opposite horizontal directions (i.e. → and ←) constitute two SPAMs. For example, SPAMs in → and ← directions are measured by $I_{i,j}$-$I_{i,j-1}$ and $I_{i,j-1}$-$I_{i,j}$ respectively. Similarly, two SPAMs for each of the vertical (↓ and↑), major (↘ and ↖) and minor diagonal (↗ and ↙) axis are calculated. A first-order Markov model utilizes these eight matrices separately to estimate their transition probabilities, i.e. probabilities of changing states in the Markov model. Among these eight transition probability tables, four tables, which are found from two horizontal (→, ←) SPAMs and two vertical (↓, ↑) SPAMs are averaged to form a single table. Likewise, the other four transition tables, which are formed from four diagonals (↘, ↖, ↗ and ↙) SPAMs, are averaged to form another transition table. A threshold T is used in the Markov model to minimize the processing complexity as well as the features in the transition table. A support vector machine (SVM) uses these two tables for both training and testing these features. Using half of the total images, at first, the SVM is trained in the features. The features of the remaining images are used to test the SVM. In the test phases, the SVM classifies an image as a stego or a cover.

The svmtrain, a MATLAB tool, is used to train the *SPAM* features of the images. The values of the parameter 'label' of the svmtrain are initialized with '0' and '1' to classify the 'cover' and 'stego' images respectively. The threshold T is set to 4 for the first order Markov model. The features of n cover images and n stego images are applied in first-order Markov model with a linear kernel. Another Matlab tool svmclassify is used to test the images and to classify these as the stego and the cover images. Total false positive (classifying cover as stego) and false negative (classifying stego as cover) are collected from the classifier confusion matrix of the tool. The average of false positive probability, P_{FP} and false negative probability P_{FN} i.e. $P_{Err} = \frac{1}{2}(P_{FP} + P_{FN})$, is used as the measure of the performance of the classifier. The values are tabulated in Table 5.4. Most of the classification errors in the first column are higher than the respective one in the second and third column. The reason is that the embedding process applied into a single error (as *nNEP*+*nPEP*=1 in the first column) will shift the pixel values in the stego image by a smaller amount. Therefore, the miss-classification rate (i.e. P_{FP} and P_{FN}) increases. Again, in the same column, the values decrease from *MLSC* to *MLDC*, *MLDC* to *MLTC* and so on because at higher cycles the shifting amount of each pixel value

increases. Consequently, the SPAM features in the stego images differ more from the SPAM features in the cover images and thus the classification error rates decrease. However, in the quadruple cycle process, the classification error increases a bit compared to the triple cycle process. This is due to the value of threshold *T*. Though many of the differences of adjacent pixels exceed [-*T, +T*] due to their larger shifting by the quadruple cycles, these do not match with any error within [-4*,* +4] during the search for the matching by the Markov model process. Thus, these do not contribute to the transition table. Consequently, the dissimilarities between the transition tables computed for the cover and the stego images are increased. A better classification accuracy is observed for *nNEP*=2 and *nPEP*=3. The phenomenon is due to the shifting of pixel values in a marginal range. The amount of dissimilarities between two transition tables increases and aids the scheme to discriminate the stego images more successfully. Again in the third column, the classification errors have been increased because the quantity of pixels, which are shifted by 4 or more, is increased, as the number of embeddable errors is 9. Thus, lot of pixel differences exceed the range of [-*T, +T*] and do not contribute to the transition table. That is why, classifier performance decreases there. Finally, it can be concluded that though fluctuations among the classification error rates are observed, the errors are closer and stay within the range [0.2, 0.27]. Thus, multi-cycle processes are not precisely differentiable from the single cycle processes by the *SPAM* feature sets.

5.5.2 Security Analysis in Generalized Benford's Law

Steganalysis based on Benford's Law is a very latest, faster and effective mechanism to detect larger modifications in a large volume of natural data. The generalized Benford's Law (gBL) is explained in Chapter 2. The successful stego detections performed by gBL in the embedding layers of 2 and 4 for each of the embedding cycles of MLSC, MLDC, MLTC, MLQC have been tabulated in Table 5.5 separately for three different image datasets. It is observed from the Table 5.5 that the detection rate increases for each higher embedding cycle without an exception in the triple cycle process. The reasons for triple cycle becoming an exception are explained in the following. In Figure 5.9, it can be verified for triple cycle embedding process at the embedding layer of 2, i.e. for 5 embeddable errors, the embedding capacity of triple cycles in BOSS, CalTech and standard images are 1.2bpp, 0.75bpp and 0.61bpp respectively. From Figure 5.12, it can be checked that single, double, triple and quadruple cycle processes provide better image quality respectively into embedding capacity range of {[0, 0.459], [0.46, 0.85], [0.851, 1.2], [1.21, 1.7]} in BOSS images, {[0, 0.245], [0.246, 0.5], [0.51, 0.855],

[0.856, 1.4]} in CalTech images and {[0, 0.21], [0.211, 0.4], [0.41, 0.8], [0.81, 1.235]} in Standard images. Therefore, at the embedding capacities of 1.2bpp in BOSS, 0.75bpp in CalTech and 0.61bpp in Standard images, triple cycle provides better image quality. That is why performance of gBL decreases in the triple cycle. Again, at the embedding layer of 4, i.e. when total embeddable errors are 9, it is noticeable in Figure 5.9 that the embedding capacities of these images are 1.4bpp, 0.85bpp and 0.8bpp respectively for BOSS, CalTech and Standard image datasets in MLTC *(i.e. k=3)*. At these embedding capacities, CalTech and Standard images provide better PSNR, as shown in Figure 11-12. On the other hand, the capacity achieved in the BOSS images in the triple cycle embedding process at embedding layer 4 is greater than 1.2. For this, the detection rate by the gBL in the triple cycle process in the BOSS image database has increased. Again, in the BOSS image dataset, achieved capacity at embedding layer 4 by quadruple cycle is about 1.55bpp, which is in the range of [1.21, 1.7]. So the detection rate decreases with the quadruple cycle.

Table 5.4: SVM's classification error rates in first four multi-cycle processes. The error rates are measured in the three image datasets by embedding into the errors for three layers of 0, 2 and 4.

Image Databases	**Schemes**	**nNEP**	**nPEP**	**nNEP**	**nPEP**	**nNEP**	**nPEP**
		0	**1**	**2**	**3**	**4**	**5**
BOSS 500	MLSC*	0.2771		0.249		0.26908	
	MLDC*	0.2711		0.24699		0.25703	
	MLTC*	0.247		0.249		0.25502	
	MLQC*	0.259		0.251		0.25904	
CalTech 5000	MLSC	0.264		0.236		0.264	
	MLDC	0.246		0.236		0.244	
	MLTC	0.234		0.238		0.256	
	MLQC	0.236		0.264		0.252	
Standard 50	MLSC	0.2581		0.2339		0.2016	
	MLDC	0.25		0.2581		0.2258	
	MLTC	0.25		0.2258		0.2097	
	MLQC	0.2258		0.2419		0.2177	

Table 5.5: Successful stego detections by gBL when each of the MLSC, MLDC, MLTC and MLQC schemes are applied in the embedding layers of 2 and 4 separately.

Image Database	MLMC Process	Successfully Stego detection			
		Total in L=2	% in L=2	Total in L=4	% in L=4
BOSS 500 images	MLSC	55	11	48	9.6
	MLDC	55	11	50	10
	MLTC	47	9.4	53	10.6
	MLQC	63	12.6	50	10
CalTech 5000 images	MLSC	490	9.8	607	12.14
	MLDC	607	12.14	617	12.34
	MLTC	529	10.58	539	10.78
	MLQC	666	13.32	588	11.76
Standard 50 images	MLSC	19	9.5	23	11.5
	MLDC	23	11.5	24	12
	MLTC	20	10	21	10.5
	MLQC	26	13	24	12

Another exception is also noticeable in the MLSC scheme in the BOSS images for L=4. In such a case, the detection rate is small because at L=4, the embedding capacity of MLSC (when k=1) is 0.4, as shown in Figure 5.9, and in that embedding capacity, BOSS images provide better PSNR for a single cycle. From that analysis, it can be concluded that embedding into multilayer makes a trade-off with multi-cycle to meet the demand for the higher embedding capacity as well as increase the stego image quality and ensure resistance again statistical attacks.

As a concluding remark, it can be mentioned that the proposed multi-cycles embedding schemes provide more resistance than single cycle schemes when the demand for embedding capacity is large. Thus, MLMC schemes exhibit more resistance against any statistical attacks.

5.6 Summary and Conclusions

Text and audio data related to investigations, interviews and various reports are usually large in volume. During the implantation of such type of massive and hybrid data, the data-hiding scheme requires large embeddable space. The proposed scheme is a pathway to fulfill such higher capacity demand. The proposed multilayer multi-cycle embedding scheme enhances

both the payload and the stego image quality compared with those of multi-layer single cycle schemes. It enhances the embedding capacity at least by a factor of 2 and several times in higher embedding cycles and layers. The scheme can conceal large volume as well as hybrid data into an image of reasonable size. It serves the demand for larger embedding capacity required for many applications including those in medical, forensic, military and law-enforcement agencies. It is also extendable to choose an arbitrary number of embeddable error points in a side in the histogram during its data implantation without any further modification in its embedding rules. The freedom of defining the number of embeddable error points in a side, choosing predictor and its parameters and selecting embedding cycles has made the proposed reversible scheme more robust and secured. Besides, it exhibits more resistance against statistical attacks when a large volume of data is embedded. For these reasons, the multilayer multi-cycle scheme is obviously a notable contribution in the field of prediction errors based reversible data hiding arena.

Chapter 6

Local Pattern Codes for Enriching Embedding Capacity

In the arena of reversible image steganography, prediction error based single layer embedding schemes implant message bits into two most frequent errors. The conceived errors are modified by one unit while implanting a bit value of 1 and remained unchanged while conceiving bit 0, or vice versa. The other errors, termed as non-embeddable errors, are shifted by one unit in the histogram by the encoder just to resolve the coinciding matter of the non-embeddable errors with the modified conceived errors. The single layer data embedment schemes try to minimize the quantity of non-embeddable errors in the embedding space for two reasons. Firstly, though the quantity of the non-embeddable errors in an image varies depending on the accuracy of the prediction process, these errors are more or less a half of total image pixels. These errors certainly destroy the originality in the stego image as they undergo a definite modification by one unit. Secondly, many of the embeddable errors are remained unchanged while conceiving a message bit of either 0 or 1, depending on the embedding rules of the applied scheme, and thus these errors lead in preserving more cover information in the stego image. These two issues imply that a method capable of generating embeddable errors, which is equal to the number of pixels in an image, i.e., no error is non-embeddable, will enhance both the embedding capacity as well as the stego image quality. In this chapter, local binary pattern (LBP) codes are generated to produce an embeddable code for each of the image pixels. Moreover, the local ternary pattern (LTP) codes are applied to breed just the required number of embeddable codes when the size of message stream is small. In case of LTP, engendered non-embeddable errors are not altered by the proposed embedding rules to improve the image quality. Both the LBP and LTP based data embedment policies are tested and compared with several latest embedding schemes [7, 31, 32, 51, 63, 83, 88, 94, 107] to justify their effectiveness and superiority over the competing ones. The proposed methods successfully pass the tests.

6.1 Introduction

During the last decade, with the enormous advances in reversible data embedment technology and its widespread applications, many researches have been proposed in the area of prediction error based data embedment process. In all the prediction error based embedding schemes, a predictor is, first, applied in image pixels to predict their values. Next, prediction errors are measured to generate the embedding space. Two or more than two highest appeared errors are chosen for secret bit implantation in the single layer (SL) and multi-layer (ML) data embedment processes, respectively. Though the ML schemes implant more bits than the SL schemes, the ML schemes destroy the image quality on a larger scale. Consequently, applications in forensic, medical and many other agencies prefer to embed into the errors of a single layer in the error space. In the SL schemes, the challenges toward enhancing the embedding capacity, mainly, lie on improving the frequency of two most appeared errors, i.e., embeddable errors. The embedding capacity means the number of implanted bits per pixel (bpp). A very common strategy of improving the number of embeddable errors is to increase the prediction accuracy. Day by day, the researchers have enriched the prediction phase of the prediction error based embedment schemes by proposing various prediction methodologies, including weighted average predictors [72, 81, 94], gradient edge detection based predictors [31, 62, 107], reference value based predictors [7, 32, 51, 59, 83] and multi-predictor based processes [11, 63]. All these prediction schemes are described in Chapter 2. The prime objective of all the prediction processes is to increase the prediction accuracy.

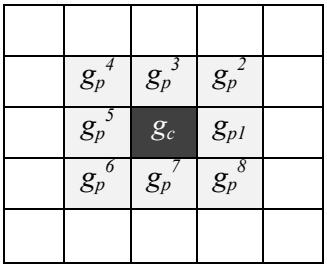
In the SL data embedment schemes, the non-embeddable errors do not accept any message bit. Nevertheless, these errors certainly destroy the originality of the cover values in the stego image because the embedding rules of these stated schemes in the immediately above paragraph shift all the non-embeddable errors by an amount to prepare space for the movement of embeddable errors. The stego image quality, therefore, mostly depends on the capability of the scheme in decreasing the quantity of non-embeddable errors; in other words, the image quality depends on increasing the number of embeddable errors. The embedding capacity, then, definitely increases. Although all the schemes try to increase the number of embeddable errors by maximizing the prediction accuracy, this is investigated that none of the schemes can bring all their prediction errors under the umbrella of embeddable errors in a single layer scheme; and even producing the exact number of embeddable errors in an image according to the requirement of embedding capacity is beyond their capability. For this reason, these SL

embedding schemes demonstrate a strong limitation in embedding into every pixel and achieving higher image quality, e.g., PSNR of more than 50 dB in all images.

In this chapter, two separate proposals are presented to meet the requirement of the smaller and higher embedding capacity by the uses of the local ternary pattern (LTP) and local binary pattern (LBP) code generation methods, respectively. Though the LTP and LBP methods are usually used in pattern matching algorithms, in this chapter, these two methods are applied for generating codes of embedding space. The LTP method computes a true value by checking whether a pixel value is less than g_c-t, greater than g_c+t or within the range [g_c-t, g_c+t], where g_c is a reference value which is a constant and t is a threshold value. Based on the true value, the LTP method produces one of the three different valued codes -1, 1 and 0, respectively, for each image pixel. Among these three codes, -1 and 1 are used for data embedment and 0 is remained unchanged by the proposed embedding rules. The LTP method controls the number of embeddable codes of -1 and 1 by repeatedly changing the values of t and produces a sufficient number of embeddable errors which are just enough for conceiving the whole message. On the other hand, the traditional LBP method generates two different valued codes 0 and 1 by comparing whether the pixel value is less than g_c or not. The embedding rules, then, implant a bit of information into all the LBP codes. Though there is no LTP and LBP based embedment scheme in the literature, for the completeness of the proposed work, several schemes that employ weighted average predictor, gradient edge detection predictor, reference value based predictor and multiple predictors for generating the embedding space are compared with the proposed LTP and LBP based schemes. The performance of the prediction processes, proposed by Tsai *et al.*, 2009 [83], Hong and Chen, 2010 [32], Hong, 2012 [31], Chen *et al.*, 2013 [11], Ou *et al.* 2013 [72], Yang *et al.*, 2013 [107], Leung *et al*., 2013 [51], Ma *et al.* 2015 [63], Chang *et al*., 2015 [7], and Wang *et al*., 2014 [94], are investigated in the experiments as the related works. The prediction processes, their limitations and the data embedment processes into these referenced schemes are summarized during their explanations in the Chapter 2. Therefore, these schemes are not restated in this chapter. The experimental results, produced by both the proposed LTP and LBP based methods, demonstrate outstanding improvement in embedding capacity and stego image quality with respect to all the competing ones.

The rest of the chapter is organized into several sections. The philosophy of using the local pattern codes LBP and LTP for the data embedment as well as their computation processes are explained in section 6.2. Section 6.3 is provided to explain the processing of the

cover image first so that the LTP and LBP based data embedment process can be implemented. Section 6.4 is devoted to showing the methodological similarities between the proposed LTP and LBP based embedding schemes and the prediction error based embedding schemes. Section 6.5 narrates the LTP code generation process and embedding method on LTP codes. Section 6.6 explains the mechanism of applying the LBP codes for data implantation. To justify the claims of obtaining better outcomes by the proposed methods, the results are demonstrated in Section 6.7. The resistance against statistical attacks is justified in Section 6.8. Finally, Section 6.9 is provided to conclude and remark the chapter.



(a)

2	5	3	9	1
6	17	29	1	5
2	5	9	9	2
3	17	3	2	9
1	2	3	2	1

(b)

If $g_p^i > g_c$
    pattern_code^i =*1*
Else
    pattern_code^i =*0*
End

(c)

1	1	0
0		1
1	0	0

(d)

If $g_p^i \geq g_c + t$
    pattern_code^i =*1*
Else if $\left| g_p^i - g_c \right| < t$
    pattern_code^i =*0*
Else if $g_p^i \leq g_c - t$
    pattern_code^i =*-1*
End

(e)

1	1	-1
0		0
1	-1	-1

(f)

Figure 6.1: Generating pattern codes for a given pattern: (a) the way of addressing contents in a pattern; (b) a sample pattern; (c) the pseudo code for LBP; (d) pattern code for LBP; (e) pseudo-code for LTP; and (f) LTP codes for *t*=5.

6.2 Generating Local Pattern Codes

Local pattern codes are used mainly for classification of objects in various retrieval purpose applications. Very commonly used local pattern matching operators are LBP, LTP, local derivative pattern (LDP), local tetra pattern (LTrP), N-th order LTrP and Gabor Transformed LTrP. While working on an image, all the processes divide the pixel grids into blocks of size 3×3 each, also known as the pattern. In each pattern, the value of the center pixel is compared with each of its surrounding eight pixels either by value, gradient directions or both depending on applied local pattern matching operator. For each comparison, a code is generated. Among these pattern-matching operators, only the LTP and the LBP produce less variety of codes. In the primary stage of the operation, the LBP generates either 0 or 1 as a code value and the LTP creates one of the codes on -1, 0 and 1 for each comparison [67]. Thus, all the generated codes in these two methods are of two and three state values, respectively. On the other hand, the LDP first computes the gradients of pixels in one of the four directions, e.g., along 0^0, 45^0, 90^0 and 135^0, and then computes a set of binary pattern codes for each of the directions. In this way, the LDP method uses more memory space and execution times while generating the same binary pattern codes. The LTrP produces one tetra pattern code of eight digits long and three binary pattern codes of 8 bits each. The computation method of the LTrP codes is more complex and it is much time consuming as well as a more space requiring process than the LBP and the LTP method. Hence, in this research, only the LBP and the LTP codes are used for the data implantation.

6.2.1 The LBP Code Generation Process

In a pattern of 3×3 sized block, the center pixel value is addressed by g_c. The other pixel values are denoted by g_p^i, where $1\le i\le8$. The value g_c acts as a reference value. The process of addressing the contents of g_p^i in a pattern is managed by the superscript value i of g_p^i, as shown in Figure 6.1(a). The LBP features are formed by comparing each of the content g_p^i in the pattern with its center value g_c. To explain the code computation method, a real image block of 3×3 pixels, as a pattern, is taken in Figure 6.1(b). The pseudo codes in Figure 6.1(c) are used to generate an LBP code for each content g_p^i of the pattern. The pseudo codes engender a code value of 1 for the pixels which are greater than the center pixel g_c and a code

of 0 for the other pixels. The generated LBP code values are displayed in Figure 6.1(d). In this scenario, the values of g_p^1, g_p^3, g_p^4 and g_p^6 are greater than g_c. Hence, a code value of 1 is generated for each of them. Code value 0 is generated from the other pixels.

6.2.2 The LTP Code Generation Process

While working on a block, the LTP code generation process first divides the grayscale pixels, values range from 0 to 255, into three parts. These three parts are 0 to $g_c - t$, $g_c - t$ to $g_c + t$ and $g_c + t$ to 255, where t is an integer value. The process generates code for each pattern value g_p^i in the pattern depending on the matching condition of whether the pattern value g_p^i is greater than $g_c + t$, less than $g_c - t$ or within $g_c - t$ to $g_c + t$. The pseudo codes in Figure 6.1(e) compute the LTP codes. The code block of Figure 6.1(e) generates a code value of -1, 0 or 1 for a pixel value of g_p^i when $0 \le g_p^i < g_c - t$, $g_c - t \le g_p^i \le g_c + t$ and $g_c + t < g_p^i \le 255$ respectively. The computed LTPs are shown in Figure 6.1(f) for t=5. In this pattern, g_p^2, g_p^7 and g_p^8 are less than g_c-5; g_p^1 and g_p^5 are within the range of [g_c-5, g_c+5]; and g_p^3, g_p^4 and g_p^6 are greater than g_c+5. For these three categories of pattern values, corresponding codes -1, 0 and 1 are generated as shown in Figure 6.1(f).

6.3 Pre-processing the Cover Image

In this phase, the image I is divided into blocks of size $d \times d$ each, where d is an integer number and the width and the height of the image I are divisible by d. It is noted that the image is divided into blocks of $d \times d$ pixels rather than of 3×3 because the proposed research uses the LBP or the LTP codes as an embedding space rather than for analysis of patterns. The LBP method generates embeddable codes for all the pixels. In the LTP method, the quantity of embeddable codes is manageable by manipulating the parameter t. Hence, the size of a block is not a concerning issue. Additionally, it creates a scope to enlarge the size of a block up to the size of the image. In each block, either the LTPs or the LBPs are computed to generate the embedding space. To compute the LBP and the LTP codes in a block, the scheme requires a reference value, g_c. The traditional LBP and the LTP methods use the center pixel of a block as the reference value for the other pixels in the block. In this case, the center pixels should be

remained unchanged, so that the decoder can generate the same pattern codes. The quantity of the center pixels is 1/8-th of the image pixels when d=3. It is possible to generate a pattern code for these block centers as well if the scheme uses a negotiated value between the encoder and the decoder as a reference value. The objective of negotiating the reference value or computing it from the block pixels is to allow the embedding process to embed into the center pixels as well as to improve the security of the scheme because without knowing the negotiated values, no third party will be able to produce the exact pattern codes. A simple method of computing a negotiated reference value for block pixels is given in the Eq. (6.1),

$$g_c = \min + \lfloor (\max - \min)/4 \rfloor * 2, \tag{6.1}$$

where min and max represent the minimum and the maximum values, respectively among the blocky pixels.

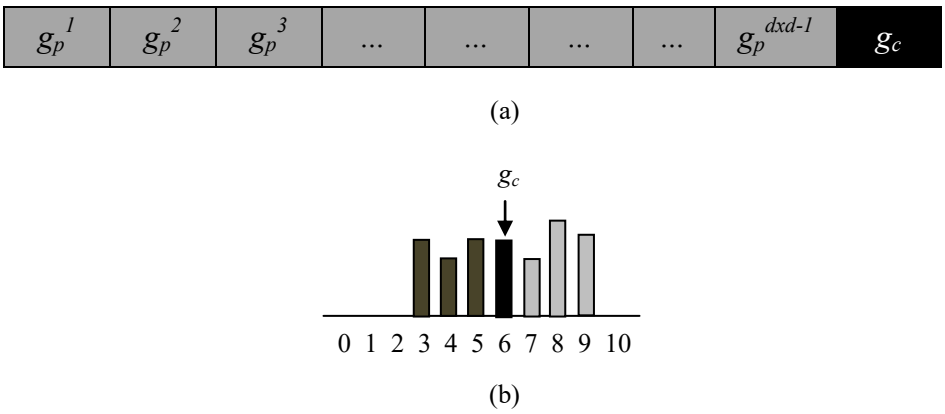


Figure 6.2: Pixels of a block: (a) array of blocky pixels after accessing through a spiral path; (b) histogram of block pixel values.

Though the reference value of a block is negotiable, for the simplicity, the center pixel of each block will be considered as the reference value, i.e., g_c, for the other pixels in the block hereafter. The pixels other than the center of the block, i.e., g_p^i, are accessed through a spiral path, where $1 \le i < d \times d$, and values are stored in an array data structure, as shown in Figure 6.2 (a) as a generalized form. A histogram of blocky pixels is drawn, as shown in Figure 6.2 (b), where g_c represents the reference value containing bin in the histogram. Regarding g_c, both the left and right side of the histogram contain odd and even valued pixels. The LBP method will be allowed to implant bits in all the pixels. The data implantation process will then modify the block pixels. In that case, the odd valued pixels could be changed to even values by the

implantation rules. The vice versa will be true as well. Many of the pixels will remain as unchanged. For example, a pixel value $I_{i,j}$ could be modified to $I_{i,j}$+1, while another pixel $I_{i,j}$+1 could remain unchanged by the data hider. That is why, a stego value of $I_{i,j}$+1 could be generated for both the cover values of $I_{i,j}$ and $I_{i,j}$+1. After the completion of the data implantation process, the decoder will be unable to detect whether the cover value of a stego pixel of $I_{i,j}$+1 is $I_{i,j}$ or $I_{i,j}$+1. To solve the problem, the proposed scheme, first, marks a bin in the histogram by the black color whose value is g_c, as shown in Figure 6.2 (b). All the pixels in the block which are smaller than the value of g_c being converted to odd numbers and the others, except the reference value, e.g. the center pixel, are converted to even values while working with the LBP method. The histogram of these converted pixel values is shown in Figure 6.3. Again, the LTP method will allow only the pixels to accept message bits which are smaller than $g_c - t$ and greater or equal to $g_c + t$. The other pixels, i.e., greater than or equal to $g_c - t$ and smaller than $g_c + t$, will not conceive any bit. That is why, while working with the LTP method, the pixels, which are smaller than $g_c - t$ and greater or equal to $g_c + t$ are converted to odd and even values, respectively, at their pre-processing stage. The other pixels, i.e., greater than or equal to $g_c - t$ and smaller than $g_c + t$, are not modified. This operation is referred in the following of this chapter as the odd-even conversion process.

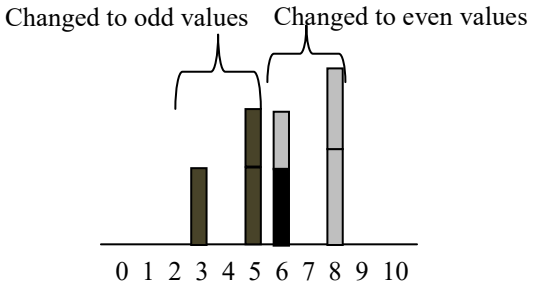


Figure 6.3: Histogram of the pixels where the pixel values less than the g_c are converted to odd values and the others are to even values.

The conversion is done based on the Eq. (6.2).

$$\tilde{g}_p^i = \begin{cases} g_p^i - \text{mod}(g_p^i, 2) + 1 & \text{if } g_p^i < g_c - t \\ g_p^i - \text{mod}(g_p^i, 2) & \text{if } g_p^i \geq g_c + t \\ g_p^i & \text{Otherwise} \end{cases} \quad (6.2)$$

where the *mod* function returns the remainder value when the first argument is divided by the second argument. The conversion equation, as shown in the Eq.(6.2), works for the LTP method. This equation will work for LBP as well for t=0. The processed pixels by the odd-even conversion process are addressed by $\tilde{g}_p^i$, as shown in the right histogram of Figure 6.3. Due to the application of odd-even conversion process, the values of pixels will be changed. These changes will create a problem in both the LBP and the LTP processes when the value of g_c in LBP and g_c+t in LTP are odd. For example, if the values of five pixels are 119, 120, 121, 121, 122, where g_c=121 and t=0, the Eq. (6.2) will change the values of 120 to 121 and 121, other than g_c, to 120 before creating LBP codes. Then the $\tilde{g}_p^i$ values will be 119, 121, 121, 120, 122. According to the demand of the embedding process, stated in the following data embedment section, the cover pixels which are greater than or equal to g_c, should never be smaller than g_c. Similarly, the pixels which are smaller than g_c should not be equal to g_c. Nevertheless, the pixel 121 is changed to 120 and the pixel 120 is changed to 121. These changes will create anomalies in the cover image reconstruction process. The problem will also be observed for t= 2, t=4, t=6 and so on in the process of LTP as the value of $g_c + t$ in these cases are odd. To solve the problem, the odd-even conversion process first checks whether the value of g_c in LBP and $g_c + t$ in LTP are odd. If these are odd valued, the value of g_c is subtracted by 1, i.e., $g_c = g_c$-1. The odd-even conversion process does not impose any restriction to the system while choosing or computing g_c so that g_c in LBP and $g_c + t$ in LTP become even valued. Rather by checking the value of g_c in LBP and $g_c + t$ in LTP, it changes the value of g_c so that g_c in LBP and $g_c + t$ in LTP become even values. All the changes that are happened by the odd-even conversion process, are traced by a binary array S_p^i, called the shift-trace array. If a change is made a bit 1 is stored in the trace array, S_p^i for that pixel. Otherwise, a 0 is stored there. An illustration of the process for a sample block is provided in Example 6.1. The compressed shift-trace array is sent to the destination through another communication channel at a suitable time. The bit stream M of the secret is then implanted into the values of pixels of $\tilde{g}_p^i$s by the proposed LTP based embedding process as described in Section 6.5 and by the LBP based embedding process as described in Section 6.6.

***Example* 6.1: *Processing a cover block* { g_p^i, g_c }={255, 250, 100, 212, 170, 131, 221, 240, 200}.**

In this example, the odd-even conversion process modifies pixels in the block. The pixels before and after modification are shown in Figure 6.4. *In this figure, the pixels in the blue cells, green cells and red cell are smaller than, greater than or equal to the* g_c (=200) *respectively. The shift-trace array* S_p^i *is used to track the changes in the pixel values by the odd-even conversion process. A* '1' *in the array indicates that a change is done to the corresponding pixel value, e.g., the first 1 indicates that the value* 254 *is a conversion value of* 255.

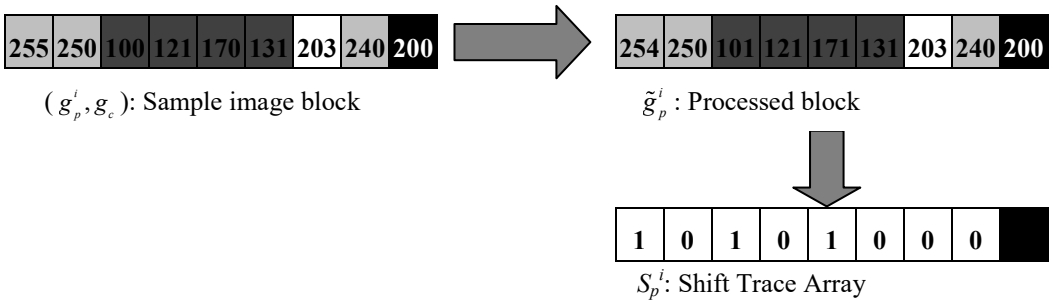


Figure 6.4: Pre-processing the cover values.

6.4 Relating the LTP and the LBP Based Data Embedment Processes with Prediction Error Based Scheme

There is no scheme in the literature that uses LTP or LBP codes for the data embedment process. To test the embedding performance of using these two proposed codes in an embedding scheme, the most commonly used prediction error based techniques are used in this chapter. As stated in Section 4.2, in a prediction error based scheme, a predictor predicts the pixel values of an image. The prediction errors are measured, thereafter and secret bits are implanted into these errors. The modified errors are added to their corresponding predicted values to form the stego pixels. Whereas, in the proposed LTP and LBP based schemes, codes

are computed first from the cover image. The difference values between the pixels and their corresponding codes are measured. These differences are addressed as the code-difference. Bits are implanted into the codes. The modified codes are added with their corresponding code-difference values to form the stego pixels. The codes and code-differences act in the proposed scheme like as the prediction errors and the predicted values. Hence, to manage terminologies in the proposed schemes more similar to prediction error based embedding schemes; these LTP and LBP code values will be addressed in the next as the encoded errors and the code-differences as the encoded prediction values.

6.5 Proposed LTP Based Data Embedment Process

In the LTP method, the frequency of 0 valued codes increases or decreases in connection with rising or falling the value of *t*, respectively. Again, the effect of *t* on the other two LTP codes -1 and 1 is opposite of the consequence on 0. This means that the frequency of 0 and the quantities of -1 and 1 are controllable by analyzing the value of *t* in the LTP method. The objective of this proposed LTP based embedding scheme is to implant bits into the error values of -1 and 1. The embedding scheme will not change the value of 0s. Hence, when the required embedding capacity is very smaller than the image size, the parameter *t* in the LTP method is assigned to a large value so that the number of 0 valued code increases and the frequencies of -1 and 1 errors become just sufficient to accept the whole message bits.

6.5.1 Generating Encoded Errors in LTP

First set the value of *t* to a small value, e.g., *t*=1. The LTP codes, i.e., the encoded errors, are computed by the methods stated in Figure 6.1(e) for all the pixels $\tilde{g}_p^i$. Let these encoded errors are e_p^i. Say, the total number of encoded errors which are valued to -1 or 1 is *L*, i.e., $L = frequency(e_p^i == -1) + frequency(e_p^i == 1)$, where $frequency(e_p^i == c)$ stands for the number of errors in e_p^i which are equal to *c*. If *L*>*M*, the value of *t* is increased by one, i.e., $t = t + 1$. The LTP codes for updated *t* and then the value of *L* are measured. If again, *L*>*M*, the value of *t* is increased by one and the LTPs and *L* computation processes are repeated. These repetitions are continued until *L*<*M*. In that stage, when it first holds *L*<*M*, the value of *t*

is decreased by one, i.e., $t = t - 1$. Finally, the encoded errors are computed with that *t*. The encoded predicted values P_p^i are computed by Eq. (6.3).

$$P_p^i = \tilde{g}_p^i - e_p^i \tag{6.3}$$

6.5.2 Data Embedment and Stego Image Generation Process

Each of the message bit *m* of *M*, where *M* is the to be embedded message stream, is embedded into the encoded errors e_p^i by Eq. (6.4). The equation generates stego-encoded errors $\tilde{e}_p^i$. Finally, the stego pixels $\tilde{\tilde{g}}_p^i$ are computed by Eq. (6.5) by adding the encoded prediction values P_p^i with the corresponding stego encoded errors $\tilde{e}_p^i$.

$$\tilde{e}_p^i = \begin{cases} e_p^i & if\ e_p^i = 0 \\ e_p^i & if\ m = 1 \\ 2 \times e_p^i & if\ m = 1 \end{cases} \tag{6.4}$$

$$\tilde{\tilde{g}}_p^i = P_p^i + \tilde{e}_p^i \tag{6.5}$$

While implanting message bit 0, the Eq. (6.4) modifies the encoded errors -1 and 1 by -2 and 2 respectively, and leaves as unaltered for embedding a bit 1. It skips the 0 valued codes as non-embeddable. The stego pixels are placed in the corresponding positions in the stego block through a spiral path. These stego blocks are concatenated to form the stego image $\tilde{I}$. The whole process is explained in Example 6.2.

Example* 6.2*: Data embedment into encoded errors

The processed block pixels $\tilde{g}_p^i$ *of Figure* 6.4 *are copied in Figure* 6.5(*a*)*. Let the embeddable message chunk is* 0011010*. This message chunk will be implanted into the pixels* $\tilde{g}_p^i$ *of Figure* 6.5(*a*)*. The encoded errors, LTP codes, are computed for t*=5 *and tabulated in Figure* 6.5 (*b*)*. The encoded predicted values are measured in Figure* 6.5(*c*) *by applying the Eq.* (6.3)*. The Eq.* (6.4) *is used to implant the message bits. Due to data implantation by the Eq.* (6.4)*, the encoded errors are modified. These modified errors are shown in Figure* 6.5(*d*)*. Finally, the Eq.* (6.5) *is applied to form the stego pixels* $\tilde{\tilde{g}}_p^i$*. The stego block is depicted in Figure* 6.5(*e*)*.*

The pixel value 203 *is not modified because the difference of it with the reference value* 200 *is less than t.*

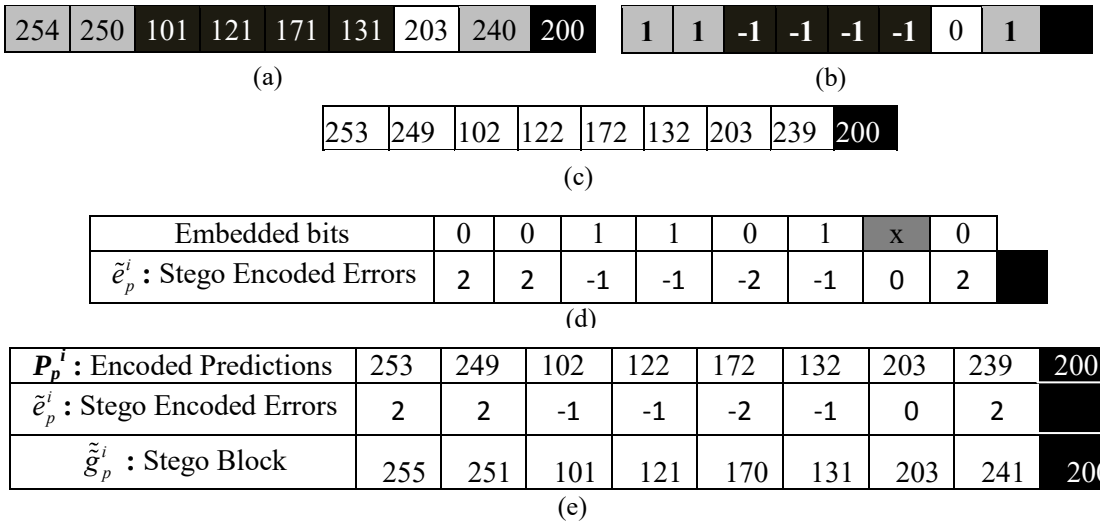


Figure 6.5: Data Embedment into a sample image block using the LTP based method: (a) processed block $\tilde{g}_p^i$; (b) encoded errors $e_p{}^i$; (c) encoded prediction values $P_p{}^i$; (d) embeddable bits and the errors after bit implantation; (e) the stego block generation process.

6.5.3 Data Extraction Process

The decoder is either informed the values of g_c and t by the encoder or able to compute these two values by using an expression. In the premier case, the encoder side sends the values of g_c and t to the decoder end by implanting these values to a separate portion in the stego image or by sending these values through another communication channel. Hence it is assumed that the decoder knows the values of g_c and t. The cover pixels which were greater than or equal to $g_c + t$, were converted to even values and the pixels which were smaller than the $g_c - t$, were converted to odd values by the odd-even conversion process, as described in Section 6.3, at their pre-processing stage. Hence, the pixels, which are smaller than $g_c - t$ and greater than or equal to $g_c + t$ in the stego block $\tilde{\tilde{g}}_p^i$ are applied in the Eq. (6.6) and Eq. (6.7), respectively, to extract each secret bit *m*.

$$m = \mathrm{mod}(\tilde{\tilde{g}}_p^i, 2) \tag{6.6}$$

$$m = \mathrm{mod}(\tilde{\tilde{g}}_p^i, 2) \otimes 1 \tag{6.7}$$

where $\otimes$ stands for bitwise exclusive or operator.

6.5.4 Cover Image Reconstruction Process

The decoder collects the shift trace array S_p^i from the encoder end through another communication channel before starting the cover reconstruction process. As the decoder knows the values of g_c and t, it is now capable to compute the encoded error values, e_p^i by using the LTP method, as shown in Figure 6.1(e). The pixels, which are smaller than $g_c - t$ were converted to even values while conceiving a bit value of 0 and remained unchanged by accepting bit 1. Hence, while implanting bit 0, the encoder changed the value of e_p^i by $2 \times e_p^i$, as shown in the Eq. (6.4). Realizing these affairs, the stego encoded errors, $\tilde{e}_p^i$ are computed at the decoder end by using Eq. (6.8).

$$\tilde{e}_p^i = \begin{cases} 2 \times e_p^i & if \ \ \mathrm{mod}(\tilde{\tilde{g}}_p^i, 2) = 0 \\ e_p^i & Otherwise \end{cases} \tag{6.8}$$

Similarly, the pixels in $_{p'}^{i}$ which are greater than or equal to $g_c + t$ are applied in Eq. (6.9) to produce the stego encoded errors, $\tilde{e}_p^i$.

$$\tilde{e}_p^i = \begin{cases} 2 \times e_p^i & if \ \ \mathrm{mod}(\tilde{\tilde{g}}_p^i, 2) = 1 \\ e_p^i & Otherwise \end{cases} \tag{6.9}$$

For the other pixels in $\tilde{\tilde{g}}_p^i$, i.e., for $g_c - t \le \tilde{\tilde{g}}_p^i < g_c + t$, the stego encoded error will be 0, i.e., $\tilde{e}_p^i = 0$.

The encoded predicted values are measured by using the Eq. (6.10). The Eq. (6.10) is inferred from the Eq. (6.5).

$$P_p^i = \tilde{\tilde{g}}_p^i - \tilde{e}_p^i \tag{6.10}$$

Finally, the pixels, which were generated by the odd-even conversion process, are measured by using the Eq. (6.11). The Eq. (6.11) is deduced from the Eq. (6.3).

$$\tilde{g}_p^i = P_p^i + e_p^i \tag{6.11}$$

The binary values of S_p^i help in identifying the pixels which were changed due to the uses of the odd-even conversion process. A binary value of 1 in S_p^i implies that the corresponding

pixel value $\tilde{g}_p^i$ is generated from g_p^i using the conversion process. Consequently, for all 1s in S_p^i, the corresponding values of $\tilde{g}_p^i$ are changed back to their cover state to form the cover pixels g_p^i. Repeating the process, all the cover blocks are reconstructed. Concatenating all these reconstructed cover blocks, the cover image *I* is retrieved by the receiver. The data extraction and the cover image reconstruction process are explained in the following Example 6.3.

Example* 6.3*: Message extractions and cover image reconstructions

Eq. (6.6) *and Eq.* (6.7) *are applied to the stego block* $\tilde{\tilde{g}}_p^i$ *to extract the message bits. Figure* 6.6(*a*) *shows the steps. In the figure, the first row provides the stego pixels. The second-row states which of the Eq.* (6.6) *and Eq.* (6.7) *are applied. The third row gives the result of these equations. These results are the extracted message bits. The pixel* 203 *is applied to none of these equations because this is within the range* $[\, g_p^i - t,\ g_p^i + t\,]$.

Figure 6.6(*b*) *explains the cover pixel generation process. In this figure, the first row represents the stego pixels. The LTP codes, i.e., the encoded errors, are generated in the second-row. Eq.* (6.8) *and Eq.* (6.9) *are applied to the stego pixels to compute the stego encoded errors. These are displayed in the third row. The first row and the third row are applied in the Eq.* (6.10) *to compute the encoded prediction values. These encoded predicted values are tabulated in the fourth row. The pixels, which were generated by the odd-even conversion process at the encoder end, are reconstructed by the Eq.* (6.11)*. These values are shown in the fifth row. The shift-trace array is found from the extracted secrets. This is provided in the sixth row. Verifying that shift-trace array, the cover pixels are retrieved in the seventh row.*

Stego Block	$\tilde{g}_p^i$	255	251	101	121	170	131	203	241	200
Applied Eq.		6.7	6.7	6.6	6.6	6.6	6.6	x	6.7	
Extracted Message	M	0	0	1	1	0	1	x	0	

(a)

Stego Block	p^i	255	251	101	121	170	131	203	241	200
Encoded Errors	e_p^i	1	1	-1	-1	-1	-1	0	1	
Applying Eqs. 6.8, 6.9	p^i	2	2	-1	-1	-2	-1	0	2	
Applying Eq. 6.10	P_p^i	253	249	102	122	172	132	203	239	200
Applying Eq. 6.11	p^i	254	250	101	121	171	131	203	240	200
Sfift-Trace Array	S_p^i	1	0	1	0	1	0	0	0	
Cover Pixels	g_p^i	255	250	100	121	170	131	203	240	200

(b)

Figure 6.6: Message extraction and reconstruction of cover pixels: (a) message extraction, (b) computation sequence during the reconstruction of cover pixels.

6.6 Proposed LBP Based Data Embedment Process

The LBP codes are used to implant data bits when the required embedding payloads M is very close to the image size, i.e., $h \times w$, where h and w stand for the height and the width of the image I. The cover image I is processed as it is discussed in section 6.3.

6.6.1 Generating LBP Codes

The odd-even conversion process is applied to compute the $\tilde{g}_p^i$ values. The LBP codes are computed from $\tilde{g}_p^i$ as it is shown in Figure 6.1(c). The LTP based data implantation rule, represented by the Eq. (6.4), embeds bits into the encoded errors of -1 and 1. The LBP method generates only the code values of 0 and 1. Hence, to embed into the LBP based encoded errors using the same embedding rules, either the embedding rules of the Eq. (6.4) have to be modified or the 0 valued LBPs have to be updated by -1. In this research, the second option is chosen so that the same embedding rules can be applied in both the LTP and the LBP based methods. Hence, after the computation of LBP codes, the 0 valued codes are modified by -1. These computed full LBP codes are regarded as the encoded errors, e_p^i. Eq. (6.3) is executed to measure the encoded prediction values, P_p^i.

6.6.2 Data Implantation Process

Eq. (6.4) is applied to implant data bits into the encoded errors. In the Eq. (6.4), the first rule, i.e., the checking of '$if\ e_p^i = 0$', is ignored. Other two checking of '$if\ m = 1$' and '$if\ m = 0$' are executed while implanting bits in the LBP based encoded errors. A flowchart of the data embedment process is shown in the Algorithm 6.1 of Figure 6.7. The algorithm is called for each of the bit implantation. To generate the pixels of the stego image block, Eq. (6.5) is executed. Finally, concatenating all the stego blocks, the stego image $\tilde{I}$ is formed.

Algorithm 6.1*: LBP_Embedment* (e_p^i , m)

Step 1. *If* $e_p^i = 1$ *then*

$\tilde{e}_p^i = e_p^i$ *and go to Step* 3.

End if

Step 2. *If* $e_p^i = 0$ *then*

$\tilde{e}_p^i = 2 \times e_p^i$ *and go to Step* 3

End if

Step 3. *Exit the module*

Figure 6.7: Flowchart of data embedment into LBP codes

6.6.3 Data Extraction and Cover Image Reconstruction Process

During the message extraction and cover image reconstruction processes, first the reference value, g_c is obtained, e.g., computed by the Eq. (6.1). If g_c is an odd value, it is reduced by one, i.e., $g_c = g_c$-1. Eq. (6.6) and Eq. (6.8) are then applied to the pixels which are less than g_c. These two equations extract the secret bits and generate the stego encoded error values, respectively. Similarly, Eq. (6.7) and Eq. (6.9) are applied to the other pixels to extract implanted bits and to produce the stego encoded errors. Eq. (6.10) is used to measure the encoded prediction values. Finally, Eq. (6.11) is applied to reconstruct the values as the odd-even conversion process generates it. The whole process is depicted in Figure 6.8. To extract each bit of the message and to reconstruct each cover pixel, the Algorithm 6.2 of the Figure 6.8 is executed. The shift-trace matrix is already discovered from the extracted bits. Checking this shift-trace matrix, the cover pixels are reconstructed.

Algorithm 6.2: LBP_Extraction (e_p^i , $_p^i$, g_c)

Step 1. *If* $_p^i < g_c$ *then*

Apply Eq. (6.6) *to compute the bit value m;*

again apply Eq. (6.8) *to compute* $\tilde{e}_p^i$ *;*

and go to Step 3.

End if

Step 2. *If* $_p^i \geq g_c$ *then*

Apply Eq. (6.7) *to compute the bit value m;*

again apply Eq. (6.9) *to compute* $\tilde{e}_p^i$ *;*

and go to Step 3.

End if

Step 3. *Compute* P_p^i *by Eq.* (6.10);

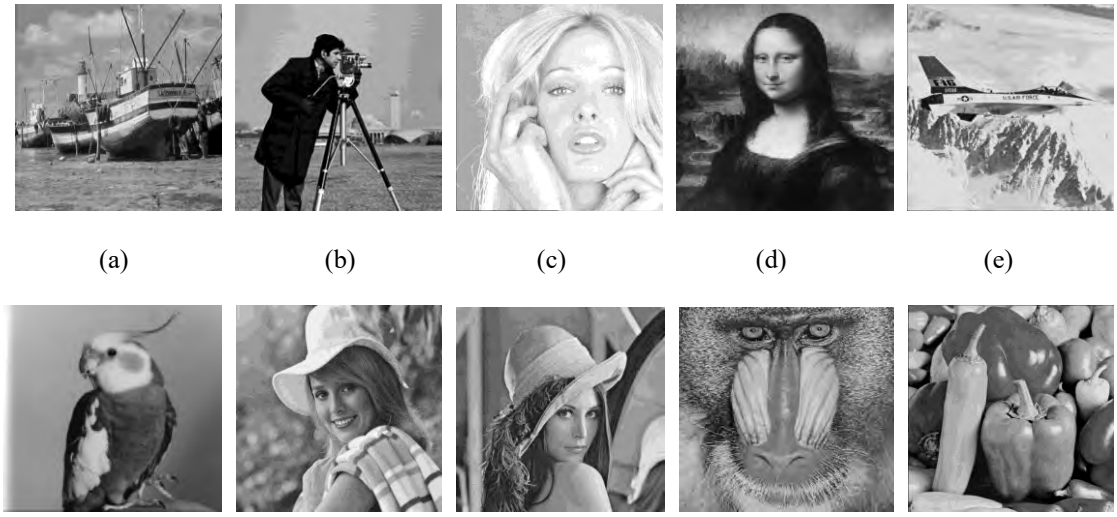
Compute $\tilde{g}_p^i$ *bu Eq.* (6.11);

Exit the module.

Figure 6.8: Flowchart to extract a message bit and to reconstruct a pixel

6.7 Result Analysis

The LBP and the LTP based embedment methods and the competing schemes are experimented on several image data sets including the CalTech, Standard and Common datasets. The results obtained by experimenting on the Common dataset are demonstrated in the following figures and tables. As a sample, very commonly used ten images of the Common data set are displayed in Figure 6.9. The proposed LBP and LTP based schemes are compared with nine reviewed schemes [7, 31, 32, 51, 63, 83, 88, 94, 107] including several latest ones [7, 51, 63, 94, 107]. The experimented schemes, their legends, which are used in the table and figures, are listed in Table 6.1. In the following discussion, the terms LBP and LTP will be used to refer to the meaning of LBP code based and LTP code based embedding methods, respectively.



(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

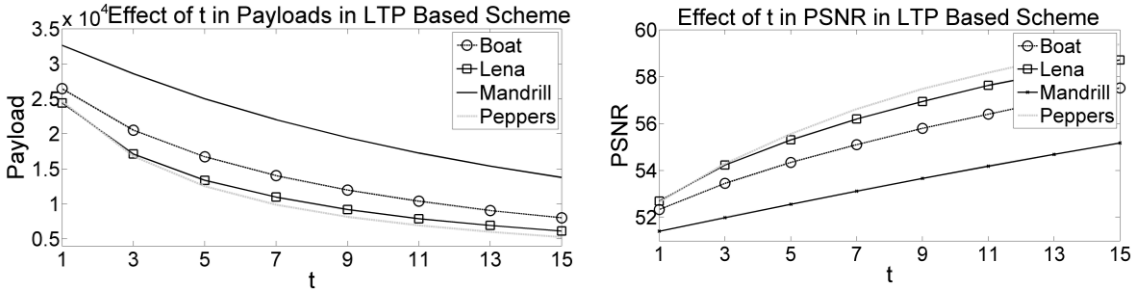
Figure 6.9: Cover images from our Common dataset: the images are (a) Boat, (b) Camera man (c) Tiffany, (d) Mona Lisa, (e) Plane, (f) Bird, (g) Elinae, (h) Lena, (i) Baboon and (j) Peppers.

6.7.1 Variation of Payload and Image Quality in LTP

Each of the LTP code is created against a specific pixel. Hence, an embeddable LTP code means that the corresponding pixel is embeddable. The quantities of embeddable and non-embeddable pixels are measurable before starting the data embedment process. The LTP method uses a parameter t to control the quantity of the embeddable codes. This method increases or decreases the number of embeddable pixels by decreasing and increasing the value of t, respectively. The method sets the value of t to the best-suited value by repeatedly varying its value in order to generate a sufficient number of embeddable codes, which are just enough to meet the embedding capacity. The objective of this analysis is to maximize the quantity of non-embeddable errors by satisfying the demand of embedding capacity. This is done because, the proposed embedding rules do not change the values of non-embeddable pixels and thus, the attempt of maximizing the non-embeddable pixels will present more quantity of not-changed pixels in the stego image. The impact of increasing the value of t on the payload and PSNR of the stego image is depicted in Figure 6.10(a) and Figure 6.10(b). In both the figures, the horizontal axis represents the t value ranges from 1 to 15. The vertical axis represents the payload and the PSNR in the Figure 6.10(a) and Figure 6.10(b), respectively. These two figures state that when the value of t is increased, the amount of payloads decreases and the PSNR value increases for all the images. The reason is that with the increasing value of t, the number of non-embeddable pixels increases.

Table 6.1: An interpretation of the legends, which are used in the figures and tables.

Sr.	Symbolic Legend	Word Legend	Status	Meaning (Scheme of)
1	P1	LBP	Proposed	
2	P2	LTP	Proposed	
3	R1	Tai	Reviewed	Tai *et al.*, [83] in 2009
4	R2	Tsai	Reviewed	Tsai *et al.* [88] in 2009
5	R3	Wien2010	Reviewed	Hong and Chen [32] in 2010
6	R4	Wien2012	Reviewed	Hong Wien [31] in 2012
7	R5	Weijen	Reviewed	W,-J Yang *et al.* [107] in 2013
8	R6	Leung	Reviewed	Leung *et al*. [51] in 2013
9	R7	Truncated	Reviewed	Chang *et al.* [7] in 2014
10	R8	Ma	Reviewed	Ma Xiaoxiao *et al*. [63] in 2015
11	R9	Wang	Reviewed	Wang *et al*., [94] in 2014



(a) (b)

Figure 6.10: Verifying the effect of *t* in LTP based method: if value of *t* increases (a) payload decreases and (b) PSNR increases.

6.7.2 Comparison of Payloads among the Schemes

The payloads of the different schemes, including LTP and LBP, for the ten images of Figure 6.9 are measured. The results are depicted in Figure 6.11 in terms of embedding capacities along the y-axis for different images. During the experiment in the LBP based scheme, the center pixels of each block of size 3x3 were regarded as reference values. Consequently, 1/9 of total pixels were not applied for bit implantation. Hence, the embedding capacity in the LBP based scheme is 0.88bpp for all the images. If calculated or negotiated values are used as the reference values, the embedding capacity will reach to 1bpp in LBP based method. In the figure, the LTP based method is depicted for *t*=3. The embedding capacity of the LTP based

scheme varies from 0.42bpp to 0.69bpp. The average embedding capacity in Ma, i.e., R8, is 0.4bpp. Average embedding capacities for all other schemes are no more than 0.2bpp because these schemes embed into their prediction errors and these applied predictors cannot produce embeddable prediction errors for all the pixels. This figure demonstrates that the LBP based scheme remarkably dominates the others. Though the embedding capacity in the LTP is smaller than the LBP, it dictates all the competing schemes in nine images out of ten.

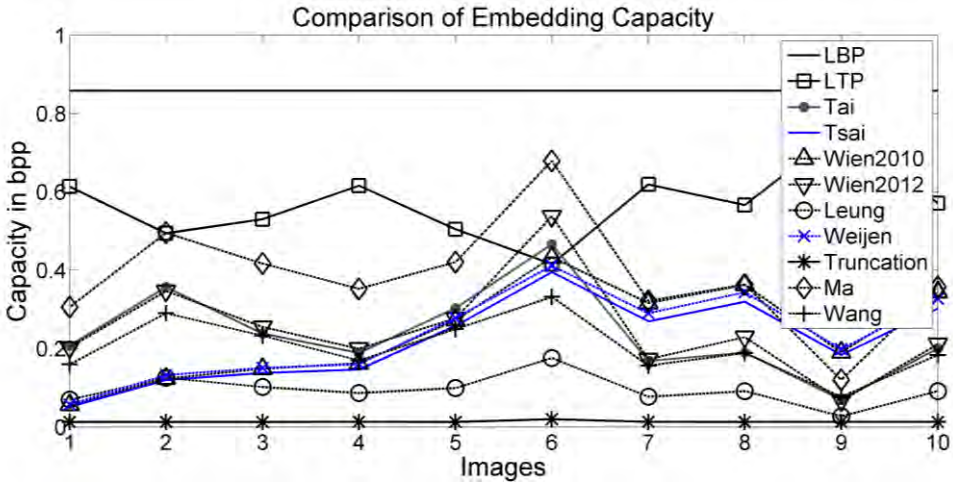
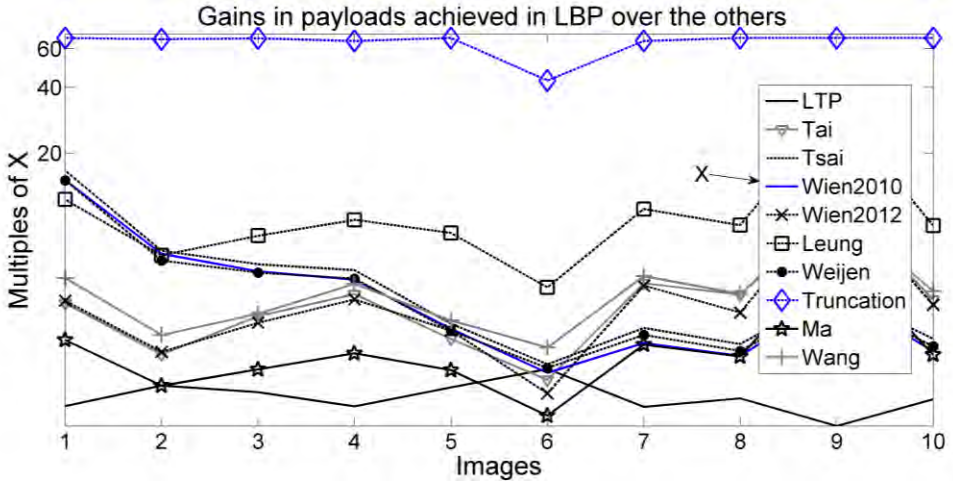
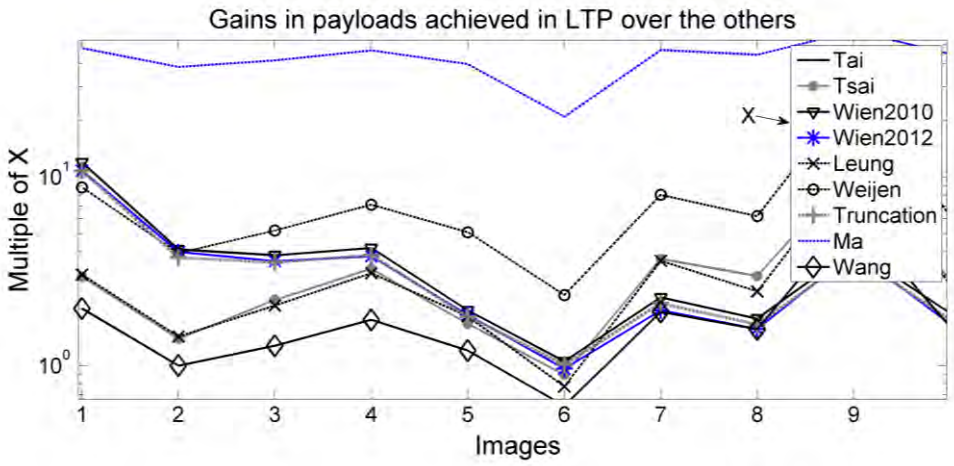


Figure 6.11: Embedding capacities which are obtained by difference schemes in the ten images



(a)



(b)

Figure 6.12: Payload gains by the scheme; (a) LBP and (b) LTP.

In another observation, payload gains in the LBP and the LTP are analyzed. To do that, the payloads of the LBP and the LTP are analyzed with the payloads of competing schemes individually for each image. These values are demonstrated in Figure 6.12 in log scale along the y-axis. The payload gain for each image is presented along with each minor tic of the horizontal axis. The gains are measured with respect to the legend stated schemes. The gains in the LBP and the LTP are presented in the Figure 6.12(a) and Figure 6.12(b), respectively. The Figure 6.12(a) states that all the gains in the LBP are positive valued. The LTP gains, shown in the Figure 6.12(b), are also positive valued other than the 6^{th} image. The positive gains indicate that the obtained payloads in the proposed LBP and LTP schemes are higher than the competing methods.

6.7.3 Comparison of PSNR among the Schemes

PSNR is used to measure the quality of an image. A higher value of PSNR indicates better image quality. Figure 6.13 is provided to depict the PSNR values of the different schemes in the ten experimented images. The experiment of LTP is conducted for t=3. Several observations that are found in the Figure 6.13 are listed in the following:

- The figure states that LTP provides best PSNR values, which are more than 56dB. This is reasoning because LTP maximizes the quantity of non-embeddable pixels for a given embedding payload and does not change the values of non-embeddable pixels. Thus, it allows a lot of pixels to remain as unchanged.

- The value of PSNR varies from image to image in LTP. This happens due to unequal payloads of LTP in the images.
- Based on the correlation of pixel values in a block, the quantity of non-embeddable pixel varies from image to image. Therefore, the images, which present higher embedding payloads for LTP method, as shown in the Figure 6.11, demonstrate smaller PSNR values in Figure 6.13.
- Though the PSNRs in LBP are smaller than the ones in LTP, these values are higher than the PSNRs of other competing schemes. The LBP provides the flat value of PSNR for two reasons. Firstly, in each image, the LBP embeds bits into all the pixels, except the block centers. Secondly, the changes of pixel values, and thus, the variations in PSNR values, are happened only for the quantity of '0' in the message stream. As the same stream is implanted into all the images, the PSNR is same for all of them, i.e., images.

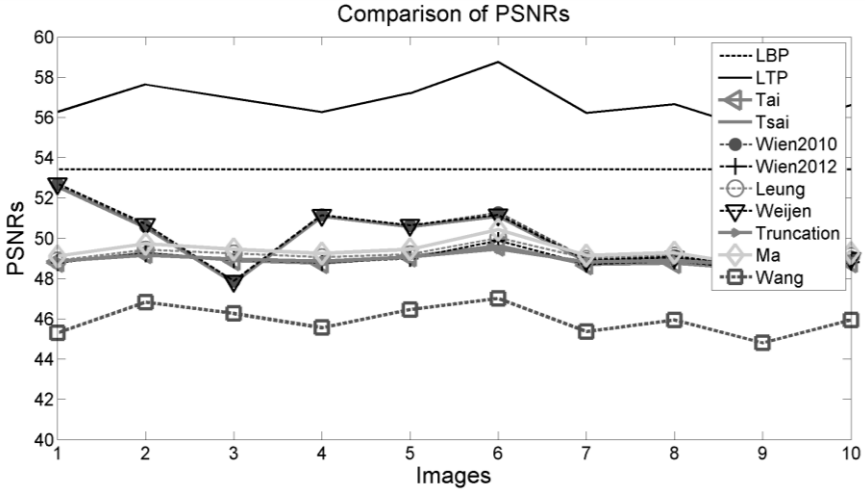


Figure 6.13: PSNR of different images in different schemes.

6.7.4 Comparison of SSIM among the Schemes

The values of structural similarity index (SSIM) are measured in the stated images for all the experimented schemes and these are depicted in Figure 6.14 along the y-axis. The figure states that the SSIM values in the LTP and the LBP are the highest. The values of SSIM in these two schemes are very close to each other. These values indicate that these two proposed schemes preserve more structural property of the cover image in the stego image. In another Figure 6.13, the payloads per structural dissimilarity index (SDIM) are sketched. The definition of

SDIM is given in Chapter 2. The payloads per SDIM in the LBP and in the LTP are very higher than the other competing schemes. Total modified pixels are less in quantity in the LTP and then in the LBP based scheme than the others. That is why these two provide better results regarding SSIM and payload per DSSIM.

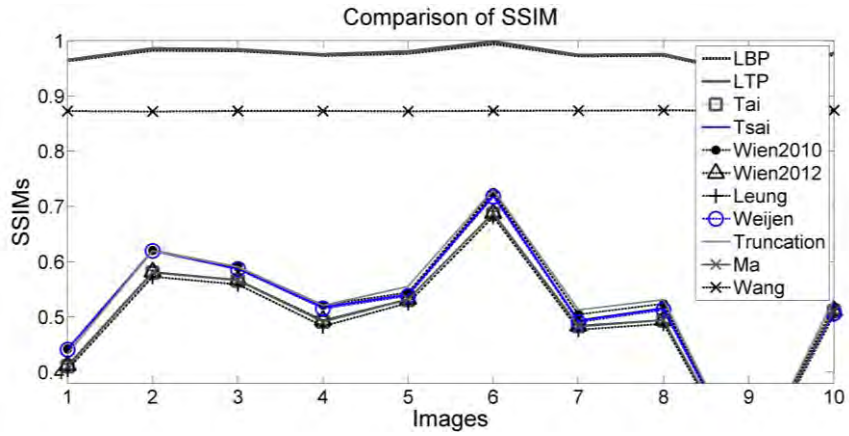


Figure 6.14: Comparison of schemes regarding SSIM in different images.

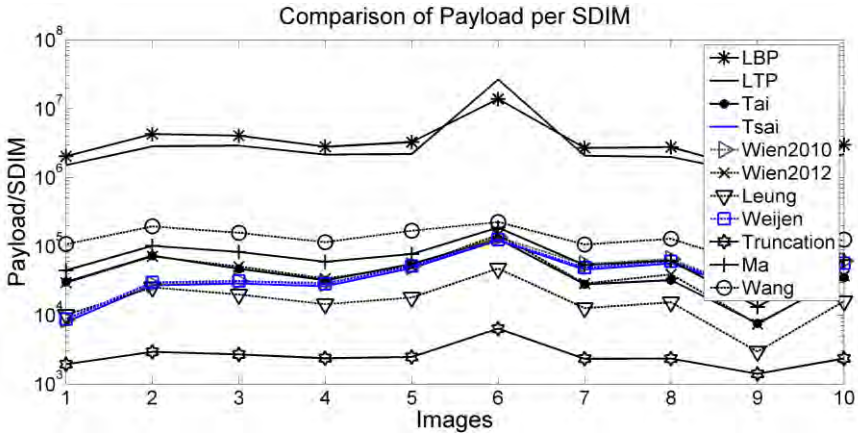


Figure 6.15: Payload per DSSIM for different images in the different schemes.

6.8 Resistance to Statistical Attacks

Due to data embedment at a rate of less than a single bit per pixel, the visual quality of a stego image degrades nominally and the changes are not detectable by the human visioning system. Nevertheless, attackers apply more insight examinations statistically to reveal the existence of hidden secrets in a carrier. To justify the resistance of our proposed schemes against attacks, several tests like relative entropy and statistical measures are conducted. As a measure of

statistical attacks, experiments on two statistical parameters like standard deviations between the cover and the stego image, correlation coefficients of them are conducted.

6.8.1 Statistical Attacks

The standard deviations are measured from the cover and the stego images in the different schemes. Minimizing the differences between the standard deviations, before and after data embedment, is a target for improving the security of the data embedment process. The divergence between these standard deviations is computed in percentage for each method. These values are shown in Table 6.2. The values in the table articulate that all the divergences, computed in different images, are less than one only in the proposed LBP and LTP schemes. These small values in the table ensure that the frequency of changed pixels in the stego image compared with the cover image in the proposed LBP and LTP schemes is smaller than the amount in the other schemes. Smaller values of these divergences, like less than 0.38% in the experimented results, will discourage attackers to look for the secret inside into the stego image.

Table 6.2: Divergence in percent between the standard deviation of the cover image and the stego image in different methods.

Image Name	P1	P2	R1	R2	R3	R4	R5	R6	R7	R8	R9
Boat	0.2	0.13	4.01	4.34	4.38	4	4	4.39	4.14	4.18	1.84
Camera Man	0.3	0.23	1.06	1.13	1.14	0.74	0.74	1.14	1.12	1.07	1.76
Tiffany	0.23	0.04	2.98	3.16	3.05	2.9	2.9	3.16	3.47	3.08	0.68
Mona Lisa	0.04	0.06	1.39	1.39	1.23	1.19	1.19	1.38	1.43	1.32	1.49
Plane	0.02	0.03	0.78	0.51	0.79	0.91	0.91	0.64	0.82	0.86	2.52
Bird	0.22	0.19	2.3	2.34	2.16	1.91	1.91	2.31	2.38	2.23	3.96
Elaine	0.07	0.05	3.29	3.29	3.16	3.18	3.18	3.3	3.28	3.17	5.5
Lena	0	0.03	1.6	1.38	1.24	1.31	1.31	1.41	1.52	1.46	1.88
Baboon	0.38	0.39	2.21	2.52	2.32	2.33	2.33	2.64	2.79	2.34	4.49
Peppers	0.01	0.01	4.26	4.2	4.05	3.93	3.93	4.18	4.21	4.09	3.36

6.8.2 Relative Entropy

The relative entropy, defined in Chapter 2, is measured by subtracting the entropy of a stego image from the entropy of a cover image. The value of relative entropy will be zero when these two images coincide. This relative entropy will be smaller when the pixel values of two different images become close to each other. The relative entropies in the LBP scheme, which are computed for different images, are shown in Table 6.3. As the payload in LBP is a flat

value, the relative entropy is computed for a single payload in an image. The highest relative entropy is 0.016, which is a very small value. In the LTP case, the entropies of the stego image for the different level of embedding payloads are measured. These values are listed in Table 6.4. It is shown from the table that the relative entropies vary within the range [0.008, 0.046]. These are also very smaller values. For three different embedding payloads, the changes in the relative entropy are 0.004 in the Bird image (measured by the difference between the highest one and the smallest one, i.e., 0.046-0.042) and less than 0.001 in all other images. These changes for different levels of embedding payloads are also nominal. Thus, the LBP and the LTP show stronger resistance against relative entropy attacks.

Table 6.3: Relative entropy between cover and stego images in the LBP based scheme

Image Name	**Implanted Data**	**Entropy in Cover**	**Entropy in LBP Stego**	**Difference**
Boat	36992	7.163	7.153	0.01
Camera Man	36992	7.024	7.011	0.013
Tiffany	36992	6.537	6.526	0.011
Mona Lisa	36992	7.486	7.481	0.005
Plane	36992	6.731	6.715	0.016
Bird	36992	7.384	7.376	0.008
Elaine	36992	7.482	7.477	0.005
Lena	36992	7.431	7.42	0.011
Baboon	36992	7.214	7.203	0.011
Peppers	36992	7.576	7.571	0.005

Table 6.4: Relative entropy between cover and stego images in the LTP based scheme

Image Name	Implanted Data	Entropy in Cover	Entropy in LBP Stego	Difference
	26451		7.148	0.015
Boat	20535	7.163	7.147	0.016
	16731		7.148	0.015
	19820		6.991	0.033
Camera Man	13250	7.024	6.992	0.032
	10779		6.992	0.032
	22863		6.515	0.022
Tiffany	14992	6.537	6.515	0.022
	10655		6.515	0.022
	26546		7.477	0.009
Mona Lisa	18970	7.486	7.478	0.008
	13841		7.478	0.008
	21761		6.703	0.028
Plane	15242	6.731	6.703	0.028
	12049		6.702	0.029
	15574		7.342	0.042
Bird	9900	7.384	7.34	0.044
	7360		7.338	0.046
	26688		7.473	0.009
Elaine	18819	7.482	7.474	0.008
	13409		7.473	0.009
	24415		7.416	0.015
Lena	17124	7.431	7.415	0.016
	13374		7.415	0.016
	32643		7.203	0.011
Baboon	28588	7.214	7.203	0.011
	25006		7.203	0.011
	24621		7.566	0.01
Peppers	16797	7.576	7.566	0.01
	12516		7.567	0.009

6.9 Summary and Comments

The proposed LBP and LTP schemes demonstrate remarkable improvement over the other competing schemes regarding all the analyzed parameters. The LBP presents the highest embedding capacity while LTP demonstrates the highest image quality. Though the image quality in the LBP is a bit lower than the LTP, this is higher than all other competing schemes. The embedding capacity in LBP is about 5 to 10 times higher than the others in an average.

The ratio of payload per SDIM in the LBP is also 10 times higher than the other schemes. The security analysis states that the LBP and the LTP based data embedment policies present stronger security for implanted data. These schemes are easier to implement and faster than the others because the methods compute a code by a single comparison. Thus, from all the analyses, the proposed LBP and LTP show better performance. It is revealed from experimental results that these two unique proposals will come in the field of covert communication as a striking methodology.

Chapter 7

Embedding by Association and Mapping of Prediction Error Histogram

Though a very common strategy of reversible data hiding schemes is to present better stego image quality, a small number of applications destroy the cover information in the stego image while implanting bits. The annihilation of cover information is intensely demanded when the cover image itself is secret. The latest policy of destroying the cover image is to apply a histogram association and mapping (HAM) strategy. In this strategy, the grayscale pixel range, i.e., 0–255, is divided into the maximum number, say k, of equally spaced partitions for each working block, so that $k \times R \leq 255$, where R stands for the range of pixel values in that block. The pixel value histogram of the block fits into one of the k partitions, known as the original partition. For each block, the scheme separates a chunk of $\log_2 k$ bits from the message stream and converts the message chunk into decimal value d. Under the reflection strategy of the embedding rules, the pixel value histogram of the block is shifted from the original to d^{th} partition and the relative position of the histogram bins, each containing a number of pixels having the same value, in the destination partition remains the same as that were in the original one. As the scheme implants $\log_2 k$ bits of information in a working block, the number of implanted bits increases for the increasing value of k, indeed for the decreasing value of R as the partitioning condition implies. In this chapter, it is proposed to apply the range of absolute values of prediction errors rather than using the range of original pixel values while implanting the HAM embedding scheme because the range of absolute values of prediction errors of a block is very smaller compared with the original value range for the same pixels of the working block and hence, the strategy improves the number of implanted bits in an image. To further improve the embedding capacity, the range of prediction errors is minimized again through applying the same predictor to the absolute values of the prediction errors and then calculating the value range of the next level prediction errors. This prediction process is repeated for n times. The proposed scheme is compared with the HAM based only existed Ong

et al.'s embedding scheme [71] and another latest image distortion based Liao *et al.*'s embedding scheme [52] to justify its superiority and effectiveness.

7.1 Introduction

In the intentional image degradation based data embedment schemes, the information of the cover image is annihilated so that no illicit person or device can retrieve the original cover contents from the stego image. Such schemes play vital roles in medical and forensic applications; in the transmission of legal documents, evidence or report by law-enforcing agencies; in communicating for copyrights and certificates; and in similar applications when the cover image itself is secret. In the intentional degradation based data embedment schemes, the image distortion is achieved by the following three ways:

i) While implanting secret bits into an embedding space, e.g., the pixel values of an image, transformed coefficients and the prediction errors of the image pixels, several embedding schemes do the data implantation task by shifting the contents of the embedding space up to a large embedding layer in their pixel value, coefficient or prediction error histogram [51]. The concepts of these schemes have been detailed in Chapter 2 and Chapter 5. When embedding is performed in the embedding space up to a large embedding layer, the pixel values are modified by a large amount and hence results in degradation in image quality.

ii) Many embedding processes, e.g., Liao *et al.* in 2015 [52] and Zhang *et al.* in 2014 [110], first annihilate the cover contents in an image by applying an encryption technique before performing the data embedment task. In such schemes, the data is embedded into the encrypted values.

iii) The pixel values of an image block are modified by an equal amount using the process of shifting all the frequency data of the pixel value histogram as a single object by an amount [71]. In such cases, the shifting amount is derived from the range of pixel values in the block.

The first methodology does not ensure the equal shifting of the pixel values. In that process, many pixels either remain unchanged or move to near values, as it is investigated in Chapter 5. Consequently, cover information is partially accessible or fully guessable. Thus, the schemes exploiting this methodology do not serve the objectives of intentionally image

degradation based data hiding policy. Though the encryption process, mentioned in the second case, fully razes the cover information, the data hiding process is not methodologically dependent on the encryption technique, rather the encryption process works independently. Thus, this is as if the schemes were embedding into a noisy image. In the third case, the pixel value histogram of a block is shifted as a single object by an amount within the gray color range. The translation of block histogram is performed by histogram association and mapping (HAM) policy. In the HAM policy, at first, a histogram for the pixels of each block is computed. The histogram is shifted to a new place in the grayscale range of 0 – 255. The value range of the block pixels and the size of message chunk to be implanted in that block determine the shifting amount. The pixel values are modified by the corresponding bin value change in the shifted histogram. This process provides a guarantee of equal modification for all the pixels of the block. More details of HAM based data embedment policy are described in Chapter 2. In the HAM scheme, the modification of pixel values is performed during the data embedment task. The target of the research, presented in this chapter, is to destroy the cover image by the embedding rules during the implantation of data bits, i.e., contributing to HAM based data embedment.

Ong *et al.* in 2014 [71] presented the HAM scheme in the field of image steganography for the first time. The presented embedding policy is chosen in this research as a benchmark for specific four reasons: (1) the HAM based embedding method is a unique one of its kind; (2) the method presents higher embedding capacity; (3) it creates larger distortions in the image; and (4) the embedding policy is quite simple. The number of bits implanted in each image block by the Ong's scheme depends on the range of pixel values in the block presented. The lower the range is higher the embedding capacity. Nevertheless, the reviewed scheme did not pay any effort to minimize the block pixel range values.

In this chapter, a novel prediction error based HAM scheme is presented where the embedding capacity is improved by applying the range value of prediction errors rather than the range of their pixel values. The motivation of using prediction errors to control the data embedment task comes from the other stego quality ensuring reversible schemes, where prediction errors are used to improve the embedding capacity. The prediction errors become very small values and are mostly close to zero. If a predictor is applied to the block pixels to predict these pixel values and the prediction errors are measured, the range of absolute values of these prediction errors will be much smaller than the range value of block pixels. In this chapter, the embedding and extraction rules of HAM policy, which were presented in the

benchmark scheme, are modified to perform HAM operation using the prediction errors. The range of absolute values of the prediction errors is employed in the proposed HAM scheme for the improvement of embedding capacity. The experimental results demonstrate an improvement over the existing HAM scheme. To further minimize the range value of block pixels, the same predictor is applied repeatedly in a way so that each time it predicts values from the lastly computed absolute values of the prediction errors, i.e., recursively applying the same predictor to the lastly computed absolute values of the prediction errors. This smaller range values help the proposed scheme in yielding a noticeable improvement in the embedding capacity.

The remainder of the chapter is organized as follows. Section 7.2 narrates the scheme of Ong *et al.* [71] briefly as a benchmark of HAM based embedding process. Section 7.3 improves the robustness of the traditional HAM scheme; while the proposed prediction error based HAM (PEBHAM) scheme detailed in Section 7.4. Section 7.5 demonstrates the experimental results and provides an analysis of the results. The effectiveness of the scheme against steganalysis is verified in Section 7.6. The proposed repeated PEBHAM (RPBHAM) scheme and the corresponding experimental results are presented in Sections 7.7 and Section 7.8, respectively. Finally, a conclusion is drawn in Section 7.9.

7.2 The HAM Based Benchmark Scheme

Ong *et al.* in 2014 [71] proposed a HAM based data embedment scheme for the first time in the literature. The scheme intentionally destroys the image contents by implanting message bits. In this scheme, the image of size $h \times w$ is first divided into non-overlapping blocks of size $m \times n$ each, where *h* and *w* are the dimensions of the image along the vertical and the horizontal directions and are divisible by *m* and *n*, respectively. While working on a block, the scheme first determines the minimum and the maximum values among the pixels. Say these are *max* and *min*, respectively. These two values are stored as the assistant information, also known as the side information, for the working block. The side information helps the receiver in its data extraction phase. The range of pixel values, say *R,* within each working block is measured by the relation $R = max - min + 1$. The grayscale range, i.e., 0-255, is partitioned into *P* parts where $P = 256 / 2^{\lceil \log_2 R \rceil}$. For $P = 4$, the gray partitions are (0-63), (64-127), (128-191) and (192-255) respectively. The value of *P* can be any of 2^i, for $1 \le i \le 8$, depending on the pixel value range of the block. Nevertheless, for the convenience, the situation of four

partitions will be exercised in all the following examples while explaining the scheme. Then, bins of the histogram of a block will be situated to a single partition as it is expected, e.g., (64-127) as shown in Figure 7.1 and the part is termed as the 'origin partition'. The block histogram situated in the original partition is moved to one of the *P* partitions as a single object by the HAM scheme during the data embedment task. These *P* partitions are known as the 'reflective partitions'. The origin partition is marked as partition no 0. Considering the grayscale as a circular path, the other partitions are numbered in an incremental way by marching towards the right direction. During the data embedment, the scheme implants *n* bits of information in a block, where $n = \log_2 P$. The value of *n* will be 2 if *P* is 4. During the data embedment task, first, the HAM scheme computes a histogram of the block pixels and associates it to the original partition. The scheme then separates *n* bits of message chunk, *msg*, from the implantable message stream and builds an association of the original partition with the partition no Bin2Dec(*msg*), where Bin2Dec converts the bit stream of *msg* into a decimal value. The partition no Bin2Dec(*msg*) is called the destination partition. The scheme next performs a bin mapping operation where each bin of the histogram, currently associated with the origin partition, is moved to the destination partition allocating the same bin position, as it was for the bin in the original partition. Stego pixels of the working block are computed from that stego histogram.

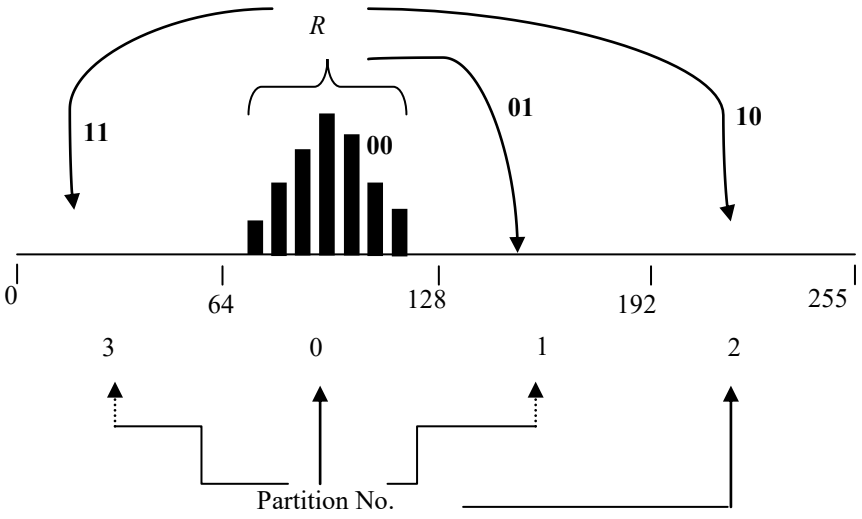


Figure 7.1: Grayscale partition and translation of histogram by HAM process.

The receiver decodes the stego image to extract the secret message as well as to reconstruct the cover image. To do the decoding operation, the receiver requires the side or assistant information that was prepared by the sender side. The assistant information is transmitted to the receiver end through some mechanisms like LSB replacement in a specific part of the host image or sending the assistant information to the receiver through another communication channel. The receiver end collects the assistant information; separates the min and max values of the cover block from the assistant information; and computes the cover block range. The receiver next computes the *min* and the *max* values of stego block as well as the stego block range. As the HAM scheme on the sender side shifted the whole histogram of pixels of the cover block to the Bin2Dec(*msg*)-th partition, the stego block range will be the same as cover block range and thus, the values of P and n are computable for the working block in the receiving end. The receiver divides the grayscale into P parts and marks the origin and destination partitions verifying the values of cover min and stego min. The partition number of the origin and the destination is then known to the receiver. Likewise the embedding phase, the receiver converts the stego block to the cover block. The extracted message chunk in the working block is Dec2Bin(destination partition no), where Dec2Bin function converts a decimal value to its equivalent binary number.

The novel HAM scheme destroys the image quality noticeably and enhances the embedding capacity. Nevertheless, the following observations state that this scheme has a good number of limitations. These are as follows:

i) The scheme uses both the *min* and the *max* values as assistant information for the respective working block and thus, has to manage and send 8x2 bits of information for each block.

ii) The final length of assistant information increases for smaller sized image blocks because the quantity of image block rises when the size of the block is reduced.

iii) After partitioning the grayscale range, the block pixel histogram may occupy the area of two partitions, e.g., if the minimum and the maximum of the block pixel is 9 and 71, the range value will be 63 and thus, the number of gray scale partition will be 4, i.e., 0-63, 64-127, 128-191 and 192-255 as shown in Figure 7.2. In this scenario, the pixel histogram occupies the region comprising of two partitions. In such context, it is not possible to define which one of these two partitions would be regarded as origin partition. This scenario remains unsolved in the scheme.

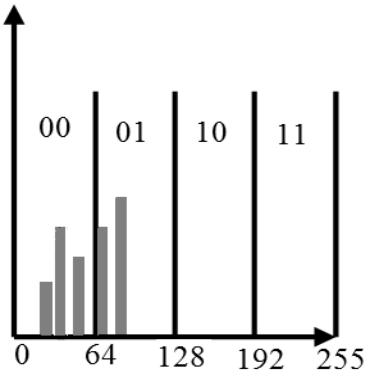


Figure 7.2: Block pixel histogram occupies the region comprising of two partitions.

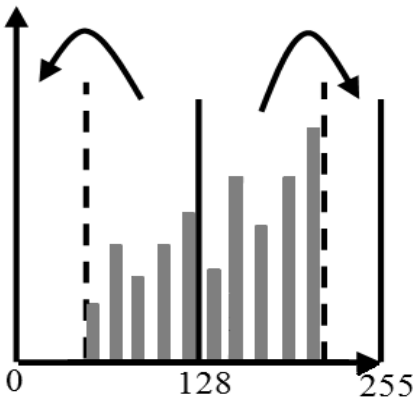
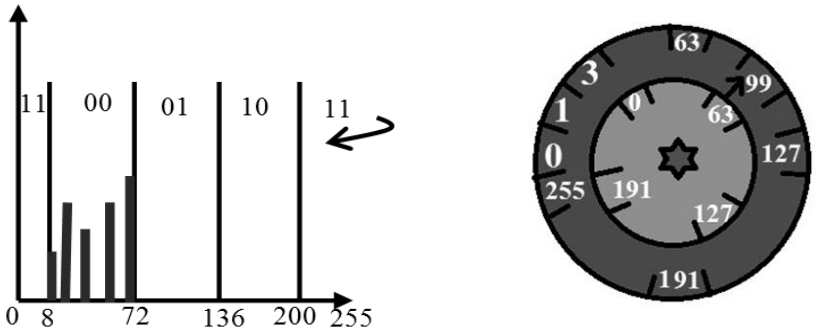


Figure 7.3: Implantation of a bit '1' when the block range is greater than 127.

iv) When the value range of block pixels is greater than 127, then there will have only one partition and bit implantation through block shifting is not possible. In such a case, the scheme proposed an alternative approach to embed a single bit only in a block. While embedding a bit '0' the block pixels are remain unchanged and for embedding a bit '1', the scheme shifts one half of the histogram bins in the right vacant places by estimating new position as x + max and another half to the left by estimating the new position as x - min, where x is the original value of block pixels. Figure 7.3 depicts the bit '1' implantation process under this scenario. However, implantation of the '1' bit using this alternative process changes the range value in the stego block with respect to the cover block. As the receiver uses the range values to extract message bits and to reconstruct the cover values, the changed range values will create ambiguity. To solve the problem, the scheme stores the *min* and the *max*

values of each block as assistant information so that the receiver can measure the range value of the cover block from the assistant information. In this process of bit implantation, the scheme introduces three new research problems. Firstly, the scheme destroys the histogram shifting property because the original histogram is not shifted equally and in the same direction; rather the scheme divides the histogram into two parts and it moves the parts in two different directions. Secondly, if the scheme could move the block pixel histogram equally in a direction without changing the range value, i.e., as it does for R≤127, only *min* value was enough to recognize the original partition and to reconstruct the cover values of the stego block. Thirdly, in the depicted scenario, many of the stego pixels will exceed the range of grayscale, e.g., if the value of *max* is 192 and the value of a cover pixel is 190, the stego pixel will be 382 which is greater than 255.

v) The embedding capacity decreases for the larger size of image blocks for two reasons– firstly, the number of image blocks decreases; and secondly, the value of each block range becomes big whereas the big value of block range decreases the quantity of implanted bits.



(a) (b)

Figure 7.4: Robust policy for HAM scheme: (a) when R≤127; (b) when R>127.

The size of assistant information stated in the limitations (i) and (ii) is remitted by considering only the *min* value of a block because only the *min* value is enough to detect its own partition. The embedding and extraction procedure is managed in such a way so that *min* value is enough to make the proposed scheme functioning. Shifting the partition walls in a direction so that a single partition can allocate all the bin values of a block pixel histogram can solve limitation (iii). The observation (iv) can be resolved by proposing a circular scale of 0 to 255 so that the stego values never can exceed the grayscale. Limitation (v) can be overcome by

proposing a new prediction error based HAM scheme. The scheme applies the range of errors to associate and map a block to a destination gray part while the gray partitions are controlled by the value of errors range.

7.3 Increasing the Robustness of the Traditional HAM Scheme

The traditional HAM scheme cannot embed in a block when the block occupies the portions of two partitions, as investigated in Figure 7.2 for $R \leq 127$ because the association of the original block cannot be defined with a single partition. Let T_s is the starting point of the partition to which the *min* value belongs to, e.g., T_s=0 in the Figure 7.2, because *min*=9 lies in the partition 0–63 as it is greater than 0 and less than 63. To solve the double partition allocation by a single block, all the partitions are shifted by L amounts, where $L = \lfloor \log_2(\min - T_s) \rfloor$. The starting point of i^{th} partition in the gray scale is marked at $T_s + L$ and end point at $T_s + L + 256/P$. The new partition for the Figure 7.2 is depicted in Figure 7.4 (a) where the partitions are redefined 9-72, 73-136, 137-200 and 201-255-8.

When $R>127$, a wheel-scale is considered in shifting the values of histogram bins to overcome the stated limitations. The process is depicted in Figure 7.4 (b). The wheel-scale is comprised of a 0 to 255 ranged outer circular scale and an inner wheel scale of the same range. The outer scale is ballasted and the inner scale is rotated for HAM purpose. Initially, the zero indicator of the inner wheel maps to the zero value of outer scale. A pointer is set to the inner wheel to indicate the *min* value of a block. The pointer moves according to the rotation of the inner wheel. The inner wheel is circulated in the clockwise direction about an amount of d to build an association of the *min* value indicator pointer with a value in the outer scale. The value of d is set to 0 for embedding a bit '0' and assigned a larger value, but less than 255, for embedding bit '1'. The value of d is a negotiable parameter. The scheme allows the data hider to choose the d value so that the image distortion is maximized. Let the pointer of inner wheel indicates a value x in the outer scale. The histogram bins of the cover block are then translated to $h(i) + x - min$, where $h(i)$ indicates the value of i^{th} bin of the cover block histogram. The translated histogram bins form the stego block histogram. Finally, the stego block is computed from that stego histogram.

The proposed scheme uses only the *min* value to implant a bit of information and does not depends on the *max* value for any of its stage of operation to decode the extracted message and

to reconstruct the cover values. Hence, the proposed scheme uses only the *min* values of blocks as assistant information. Thus, the scheme reduces the size of the assistant information into a half.

The adopted strategies solve the stated limitations (i)–(iv) and have been published in the Journal of Computer Science [29]. The scheme increases the embedding capacity, however, it does not take any measure to pre-process the image block for decreasing the value of block range. Section 7.4 modifies the proposed scheme further through the inclusion of prediction error based histogram shifting strategy to achieve higher embedding capacity by ensuring the smaller sized image blocks.

7.4 Proposed PEBHAM Scheme

In the PEBHAM scheme, a predictor is used to estimate the pixel values of the k-th block, B^k. The scheme measures the range of the absolute-valued prediction errors. This range of the block is used to translate its block histogram, indeed shift the block pixels, while implanting a message chunk. For the purpose of the message extraction and the cover images reconstruction, the same predictor is applied at the receiver end. The proposed scheme requires that the prediction error values in the cover image should be equal to the corresponding prediction error values in the stego image.

7.4.1 Predicting Block Pixels

The target of this section is to apply prediction errors to implement the HAM scheme. For the operational purpose of the proposed PEBHAM scheme, the predictors are chosen so that the prediction errors in the cover block and that in the stego block are the same. A mean value predictor is very suitable for generating the same prediction errors in the cover block and also in the stego block because the HAM policy shifts the pixels of the cover block by an equal amount in a direction. Though there are many mean value predictors in the literature, for the robustness of the scheme, a new prediction policy is proposed in this chapter. In the prediction method, the corner pixels, the edge pixels and the inner pixels of a block are estimated by the mean of two, three and four neighbor pixels, respectively. The inner pixels, the corner pixels and the edge pixels are shown in Figure 7.5(a) by filling them with white color, gray color and black color, respectively. A snapshot of associated neighbor pixels, while estimating the corner

pixel *a*, the edge pixel *b* and the inner pixel *g* are shown in Figure 7.5(b), Figure 7.5(c) and Figure 7.5(d), respectively. The values of *a*, *b* and *g* are predicted by using the Eq. (7.1), i.e., using Eq. (7.1.1), Eq. (7.1.2) and Eq. (7.1.3) respectively.

$$\left.\begin{aligned} P_a &= (b+f)/2 && (7.1.1)\\ P_b &= (a+c+g)/3 && (7.1.2)\\ P_g &= (b+h+l+f)/4 && (7.1.3) \end{aligned}\right\} \quad (7.1)$$

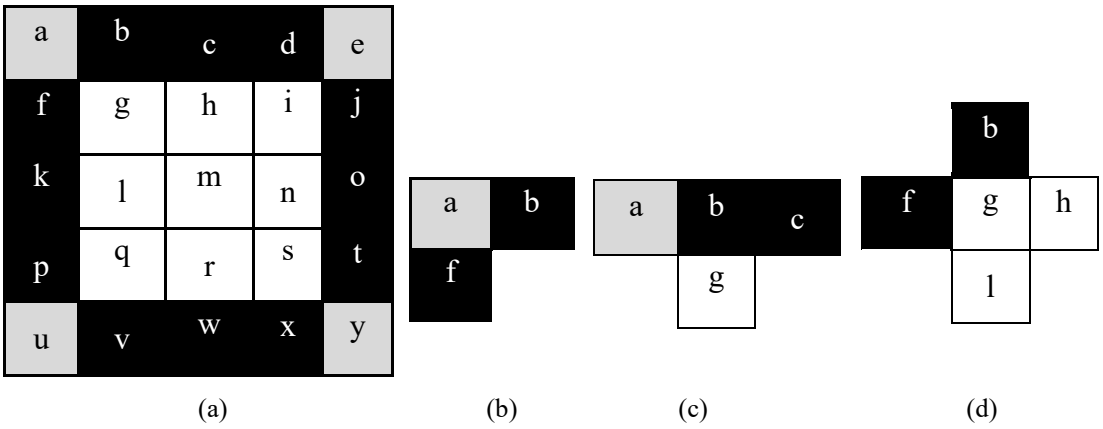


Figure 7.5: Prediction process of cover block: (a) a cover block (b) pixel value *a* is predicted by *b* and *f*; (c) *b* is predicted by *a*, *c* and *g*; and (d) *g* is predicted by *b*, *f*, *h* and *l*.

The prediction errors $E_{i,j}$ are measured by subtracting the predicted values from the corresponding pixel values. The absolute values of the prediction errors are computed by Eq. (7.2).

$$E^{A}_{i,j} = ABS\left(E_{i,j}\right) \quad (7.2)$$

Where the function *ABS*(.) returns a positive value irrespective of whether the value of $E_{i,j}$ positive or negative.

7.4.2 Computing Error Range

The value range of $E^{A}_{i,j}$ in a block is measured by $R_e = \max Val\left(E^{A}_{i,j}\right) - \min Val\left(E^{A}_{i,j}\right) + 1$, where the *maxVal* and the *minVal* functions return the maximum and the minimum values, respectively, from the absolute errors in $E^{A}_{i,j}$. This range value R_e is smaller than *R*. The value of R_e is used for dividing the gray scale pixel range. In the HAM scheme, when *i* bits of information is implanted into a block, the gray scale is partitioned into 2^i parts so that a

partition can be chosen for any possible pattern of i bits. The scheme embeds up to 8 bits in a block, as the grayscale range is 0-255 and 2^8=256. The number of bits that are embedded in a block varies for the value of block range R_e. The length of message chunk, i, is determined by $8-\lceil \log_2 R_e \rceil$. The partitioning value R_p and the number of partitions P are computed by using Eq. (7.3) and Eq. (7.4), respectively.

$$R_P = 2^{\lceil \log_2 R \rceil} \tag{7.3}$$

$$P = 256 / R_p \tag{7.4}$$

The grayscale partitions for a sample value of R_P=32 is depicted in Figure 7.6.

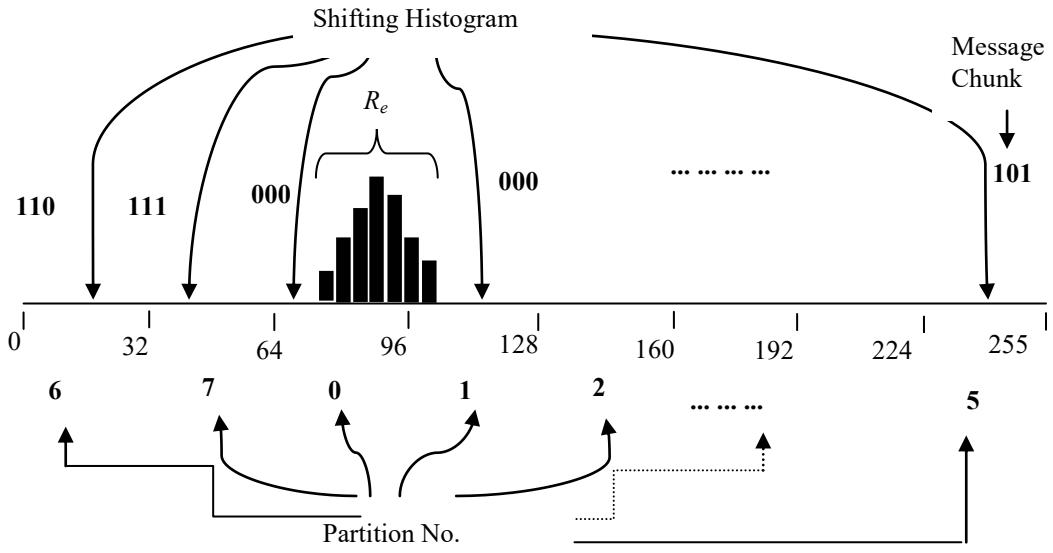


Figure 7.6: Prediction error based HAM scheme

7.4.3 Data Embedment Process

Let the minimum value of the pixels in a block is Min_c. The value Min_c belongs to a partition in the grayscale. This partition is numbered as '0'. A partition labeler marches to the right direction numbering each partition in an incremental order of 1, 2, 3, ..., until it reaches the last partition. Thereafter, it moves to the leftmost partition and again moves to the right direction until it labels '(P-1)', i.e., before the start of the partition numbered ‘0’. In Figure 7.6, the partitions are labeled as 6, 7, 0, 1, 2, 3, 4 and 5. In each block, a message chunk of 3-bits can be embedded. The number of messages bits b to be embedded into a block is determined by

the equation $b = log_2 P$. Let m be the b-bits message chunk. The binary message chunk m is converted to decimal value d. The Min_c value of the working block is associated and mapped to the partition number d. The stego block S^k is formed using the Eq. (7.5), where T_0 and T_d are the starting value of partition no 0 and d, respectively. Several pixels in S^k may exceed the extreme gray value 255 if $\max + T_d - T_0 > 255$, as it is depicted in Figure 7.7 (a). Hence the final stego block $\tilde{S}^k$ is formed by flipping these extreme values using Eq. (7.6). The stego block histogram after applying the Eq. (7.6) is depicted in Figure 7.7(b).

$$S^k = B^k + T_d - T_0 \tag{7.5}$$

$$\tilde{S}^k = \mathrm{mod}\left(S^k, 256\right) \tag{7.6}$$

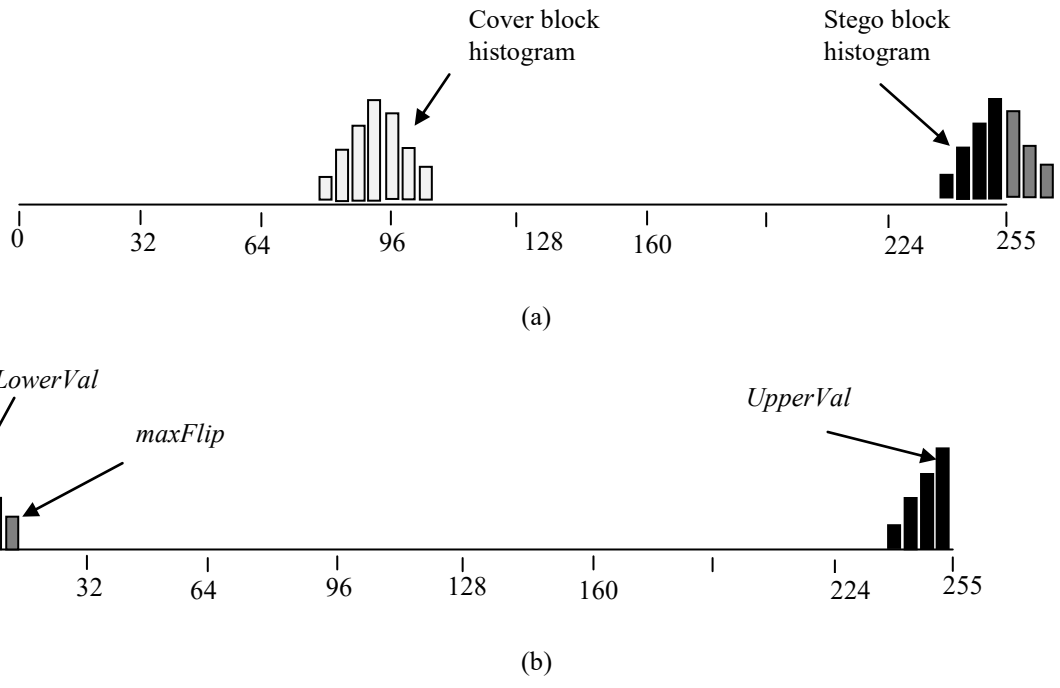


Figure 7.7: Stego block generation: (a) Stego block histogram after applying the Eq. (7.5); and (b) Stego block histogram after applying the Eq. (7.6).

When the Eq. (7.6) is applied, a portion of the stego pixels of the stego block will reside very near to 255 while the others will take place very close to 0 due to the flipping operation. When the stego pixels are found at the two extremes of the gray scale, i.e., near to 0 and 255, it infers that a flipping operation was executed by the Eq. (7.6). The embedding algorithm measures some additional information so that the data extractor can find out on which shifted

values the flipping operation has been applied. It measures the maximum value among the pixels of $\tilde{S}^k$ in each stego block when a flipping operation is realized. The minimum one of these maximum values, regarding the whole blocks, is stored in a variable named *UpperVal*. Similarly, the scheme measures the maximum of the minimum values of only the flipped pixels in the stego image. Let, the value is *LowerVal*. The maximum of the flipped pixels, regarding the whole stego pixels, are also measured. That flipped maximum value is addressed by *MaxFlip*. These *LowerVal*, *UpperVal* and *MaxFlip* are made part of side-information. Regarding the whole image, 24 bits of side information is required for these three values. The first two parameters are used to estimate whether a block undergoes a flipped operation or not. If a flipping operation is detected in a block, the last parameter is used to find the pixels on which the data hider performs the flipped operation. All the stego blocks $\tilde{S}^k$ are concatenated to form the stego image $\tilde{I}$. The embedding process is demonstrated in the Example7.7.

7.4.4 Side Information

For each block, as a side information, the competing scheme [71] uses the minimum and the maximum value of the block and the number of shifted pixels in the left and the right side of the block pixel histogram when R>127. The scheme added 32-bits to the side information for each block to manage these four values. The proposed scheme manages only 8 bits of side information for Min_c for each block. In addition to that, the scheme stores 24-bits more at the final step, i.e., 8 bits for *LowerVal*, 8 bits for *UpperVal* and 8 bits for *MaxFlip*.

The sender sends the side-information to the destination either by embedding these as a part of the cover image or by sending them separately through the alternate mechanism. The management of the side-information, including the selection of parameters and allocation of bits for each one, compression techniques, a mechanism for sending these to the destination, etc., are described in many of the literature including [32, 51, 57, 94]. The scheme presented in [71] has not described the policy of transmitting the side information. Therefore, the proposal is also avoiding the communication process of side-information because by this time the length of side information is minimized and thus, there is no way of degrading the performance level compared to its competing schemes for the maintenance of side-information.

7.4.5 Data Extraction and Cover Image Reconstruction Process

The decoder extracts the values of Min_c, $LowerVal$, $UpperVal$ and $MaxFlip$ from the side information. The minimum and the maximum values of the stego block is measured by *min* and *max*. If $min \le LowerVal$ and $max \ge UpperVal$ at a time, a flipped operation, that was performed by the Eq. (7.6) on the sender side, is detected. In that case, some of the pixels require being flipped back to the state of Eq. (7.5). During returning to the state of $S_{i,j}^k$ from the state of $\tilde{S}^k$, the flipped pixels are determined. In the flipped block, the flipped pixels are these pixels which are less than or equal to $MaxFlip$. These pixels are summed by 256 to construct the $S_{i,j}^k$. The construction of $S_{i,j}^k$ is summarized as an expression in Eq. (7.7).

$$S_{i,j}^k = \begin{cases} \tilde{S}_{i,j}^k + 256 & \text{if } \tilde{S}_{i,j}^k \le MaxFlip \\ \tilde{S}_{i,j}^k & \text{Otherwise} \end{cases} \tag{7.7}$$

The scheme now applies the same prediction policy on $S_{i,j}^k$, as it was applied by the encoder, and measures the values of $E_{(i,j)}^A$, R_P and P. The present stego displacement dS^k in $S_{i,j}^k$, regarding the cover block, is measured by the Eq. (7.8). The cover block is reconstructed in Eq. (7.9) by again subtracting the amount of stego displacement from the $S_{i,j}^k$. The decimal value of the secret is found by the Eq. (7.10).

$$dS^k = \min\left(S_{i,j}^k\right) - Min_c \tag{7.8}$$

$$B_{i,j}^k = S_{i,j}^k - dS^k \tag{7.9}$$

$$d = 2^{\left\lfloor \log_2 (dS^k / R_P) \right\rfloor} \tag{7.10}$$

Finally, *d* is converted to binary value *m* and sufficient '0's are appended to the left of *m* to make it *b*-bits length. The whole process is demonstrated in Example 7.2.

The process is repeated for the other blocks in the stego image. The proposed work is published in the proceedings of International Conference on Networking Systems and Security (NSysS) in 2016 [42].

7.4.6 Solving the Flipping Detection Anomalies for the Wider Block Range

The pixel values of a block do not change for conceiving a 0 valued bit. When the range of pixel values of a block is very closed to 255 and the sender side implants a bit 0 into the block, the receiver end will be misguided to detect it as a flipped block because in this stego block the relationships of $min \leq LowerVal$ and $max \geq UpperVal$ hold at the same time. The problem is solved in the proposed scheme by assigning 255 to the side information parameter Min_c. In such a block, the bit value of 0 is not implanted and the block is remained unchanged by the sender side. Therefore, in that case, Min_c does not contain the *min* value of the block pixels, rather it is fixed to 255 as an indication mark of the skipped block. The blocks that contain all 255 valued pixels are also avoided from any bit implantation because in these blocks Min_c=255. When the receiver end detects that Min_c=255, it avoids the data extraction and cover block reconstruction processes, as the block was ignored for bit implantation by the sender side.

Example 7.1: Implanting a message stream *m* = 00101 into an image block, as shown in Figure 7.8 (a).

The block as shown in Figure 7.8(*a*) has a $Min_c = 206$. *After the prediction process, the predicted values, the prediction errors and their absolute values are demonstrated in the tables of Figures* 7.8(*b*), 7.8(c) and 7.8(*d*), *respectively. It is found that* R_e=7 *and hence,* R_p=8, P=32 *and* b=5. *Let the decimal value of pointed message chunk m is* 5. *According to the stated procedure, the partition number* 0 *is* 199 *to* 206. *The value of* Min_c *is associated and mapped to the partition number* 5 *having the pixel value range of* 239 *to* 246. *Hence,* T_0 =199, T_d =239 *and* T_d - T_0 =40. *According to the Eq.* (7.5), $S^k = B^k + 40$. *The block* S^k *is shown in Figure* 7.8(*e*). *Finally, the flipping operation is performed by the Eq.* (7.6). *The final stego block is demonstrated in Figure* 7.8(*f*). *As the side information,* Min_c, $LowerVal$, $UpperVal$ *and* $MaxFlip$ *values are updated. Note that,* $LowerVal$, $UpperVal$ *and* $MaxFlip$ *are updated for the whole image while* Min_c *is stored for each block. Regarding this block,* Min_c =206, $LowerVal$ =1, $UpperVal$ =255 *and* $MaxFlip$ =9.

225	222	220
217	215	211
212	210	206

(a)

219	220	216
217	215	213
213	211	210

(b)

6	2	4
0	0	-2
-1	-1	-4

(c)

6	2	4
0	0	2
1	1	4

(d)

265	262	260
257	255	251
252	250	246

(e)

9	6	4
1	255	251
252	250	246

(f)

Figure 7.8: Data embedment into an image block: (a) image block; (b) the predicted values; (c) the prediction error *E*; (d) absolute error values $E^A_{i,j}$; (e) results of the Eq. (7.5) $S^k = B^k + T_d - T_0$ (7.5); and (f) the stego values computed by the Eq. (7.6).

Example 7.2: Message extraction and cover block reconstruction

The decoder end first collects the values of Min_c =206, *LowerVal* =1, *UpperVal* =255 *and* *MaxFlip* =9 *from the side information. The stego pixels are depicted in Figure* 7.9 (*a*). *The min and max values are* 1 *and* 255 *respectively. Both the conditions of* $min \le LowerVal$ *and* $max \ge UpperVal$ *are satisfied. This means that it is a flipped block. The pixel values smaller than or equal to* 9 *are flipped back by adding* 266 *to them. The flipped back stego block* S^k *is depicted in Figure* 7.9(*b*). *The predicted values of the pixels and the prediction errors are presented in Figure* 7.9(*c*) *and Figure* 7.9(*d*), *respectively. From these prediction errors, the values of* $E^A_{(i,j)}$, R_P P *and b are measured. The stego displacement* dS^k *is computed by applying the Eq.* (7.8). *Finally, the Eq.* (7.9) *is employed to reconstruct the cover block, as shown in Figure* 7.9 (*e*). *The message in decimal format is extracted by using the Eq.* (7.10). *The computed values of b and d are* 5 *and* 5, *respectively. The value of d is converted to binary, which is* 101. *As the length of message chunk is* 5, *because of b =* 5, *two zeros are concatenated at the begging of the binary string to form the message chunk m. The extracted message chunk is* 00101.

9	6	4
1	255	251
252	250	246

(a)

265	262	260
257	255	251
252	250	246

(b)

259	260	256
257	255	253
253	251	250

(c)

6	2	4
0	0	2
1	1	4

(d)

225	222	220
217	215	211
212	210	206

(e)

Figure 7.9: Message extraction and cover block reconstruction: (a) stego block; (b) flipped stego block yield by applying Eq. (7.7); (c) predicted stego block; (d) stego prediction errors; (e) cover block constructed by Eq. (7.9).

7.5 Result Analysis

The experiments are performed using 5000 CalTech images, 500 BOSS images and 50 standard images. The images are resized to 240x240 before applying the embedding process. After the data embedment task, the embedding payload and the peak signal to noise ratio (PSNR) are measured. These values are investigated and analyzed for various sizes of blocks. The results are compared with the benchmark scheme Ong *et al.* in 2014 [71]. To the best of the author's knowledge, there is no other HAM based data embedment scheme in the literature to compare the results of the proposed scheme. Nevertheless, the scheme of Liao *et al.* [52] is incorporated in this section to compare its results with the proposed scheme because, though the scheme is not similar, like the proposed one, it destroys the image quality intentionally. Additionally, the proposed scheme uses the absolute values of prediction errors in a state of the embedding procedure, while the scheme of Liao *et al.* uses the absolute difference of neighbor pixels to control their data embedment process. These two schemes relate themselves by using absolute operations in their data embedment and extraction processes. The results obtained from different schemes are demonstrated in the following. The demonstrated results justify the claim of boosting up the embedding payloads and improving the level of image distortions by the proposed scheme.

7.5.1 Justification for Applying the Error Range in HAM policy

To demonstrate the justification of using error range in HAM scheme rather than using the range of pixel values, the prediction errors are measured by applying the proposed predictor in a sample image of CalTech dataset. The results are analytically observed in Figure 7.10, where the histogram of the prediction errors and the pixel values are drawn in the same plane. Most of the prediction errors are condensed to '0' or close to '0', whereas the pixel values are distributed over the gray range at about a flat rate. Which implies that the pixel values in the blocks are distributed in a wider range. The range value R of pixel values in a block becomes larger. On the contrary, prediction errors are distributed into a smaller range. As a result, the range value R_P of the prediction errors in a block becomes smaller.

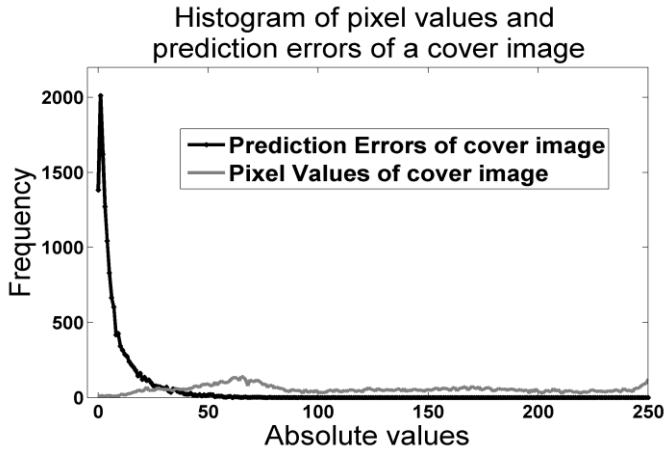


Figure 7.10: Histogram of pixel values and prediction errors computed from an image of the CalTech image dataset.

Table 7.1: Average number of blocks whose block range is greater than 64 at various block sizes

Image Database	Scheme	Block Size			
		3x3	5x5	8x8	12x12
BOSS	Ong *et al.*	435	312	189	115
	Proposed	110	69	41	27
CalTech	Ong *et al.*	1266	776	416	240
	Proposed	388	236	137	80
Standard	Ong *et al.*	1136	801	455	249
	Proposed	198	126	77	49

The embedding capacity decreases for large valued ranges. When the value of R in a block is greater than 64, the number of gray partitions P will be 2 because $P = 256 / 2^{\lceil \log_2 R \rceil}$. The competing scheme will then be able to implant only a single bit in the block as $b = log_2 P$. It is true for the proposed scheme as well when R_p is greater than 64. The value of R_p in the prediction errors and the value of R in the pixel values of each block are measured. The measurements are done in three different image categories and various sizes of the image block. The number of blocks, each having a pixel value range of greater than 64, are counted and tabulated in Table 7.1 separately for [71] and the proposed scheme. It is investigated that the quantity of such large ranged blocks in [71] is about 3 to 6.36 times compared with those found in the proposed scheme.

7.5.2 Analysis of Embedding Payloads

The payloads of the proposed scheme and its competing schemes are measured in different image datasets. Only the results obtained in the Caltech image dataset are demonstrated in Figure 7.11. The payloads of the first 100 images of the CalTech image dataset are depicted along the y-axis.

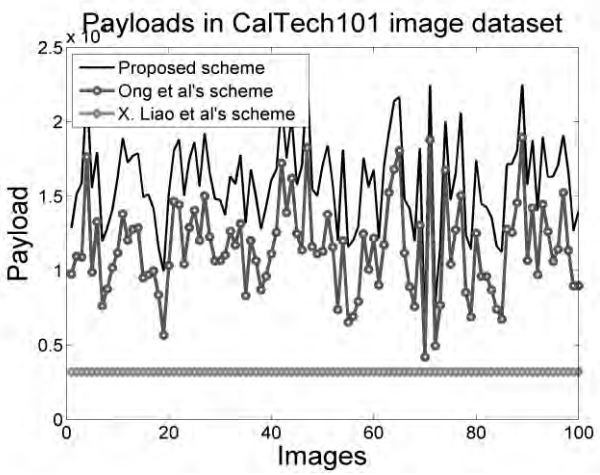


Figure 7.11: Comparison of payloads between the proposed scheme with the schemes of Ong *et al* and Liao *et al*. The results are presented from the first 100 images of the CalTech image dataset.

Figure 7.11 states that the proposed scheme dominates the others noticeably by the achieved payloads. Among the compared schemes, Liao *et al.*'s one [52] presents the lowest embedding payloads because in this method only a single bit of information is implanted in each image block. Ong *et al.*'s scheme [71] presents a bit improved payloads due to their attempts of embedding multiple bits in each image block. The proposed method improves Ong *et al*.'s embedding payloads by a factor of about two. The result is achieved by applying the ranges of the absolute values of the prediction errors while implementing the HAM operations. As only Ong *et al.*'s scheme competes with the proposed scheme by its payloads, these two schemes are further compared in Figure 7.12 for different sizes of image blocks. The average payloads in each image datasets are drawn along the y-axis and the results against different block sizes are shown along the x-axis. The results of the proposed scheme and the Ong *et al*.'s scheme are grouped in the figure for each block size. The embedding payloads in the proposed scheme are much higher than that obtained in Ong *et al*.'s scheme for all sizes of image blocks. The payloads for a block size of 3x3 is more than the payloads in the other block sizes because, in the block size of 3x3, the number of processed blocks are more in quantity than the others and

data are embedded into each block separately. With comparison to Ong *et al.*'s scheme, the average payloads increases in the proposed scheme in the image dataset of BOSS, CalTech and Standard by a factor of {1.31, 1.51, 1.58}, {1.33, 1.62, 1.77}, {1.35, 1.73, 1.93} and {1.38, 1.83, 2.04} respectively for image block of size 3x3, 5x5, 8x8 and 12x12. As the factors are greater than 1, it certainly proves an improvement. It is also inferred that the proposed scheme improves its payloads by a factor of 1.31 to 2.04 depending on the categories of images and sizes of image blocks.

The minimum payloads that are obtained in each image dataset are also compared for different sizes of image blocks in Figure 7.13. There, it is noticeable that the proposed scheme dominates the competing one by a factor of more than 2. Thus, it can be concluded that the proposed scheme definitely enhances the embedding payloads and this improvement is always obtainable irrespective of the sizes of blocks and the sources of images.

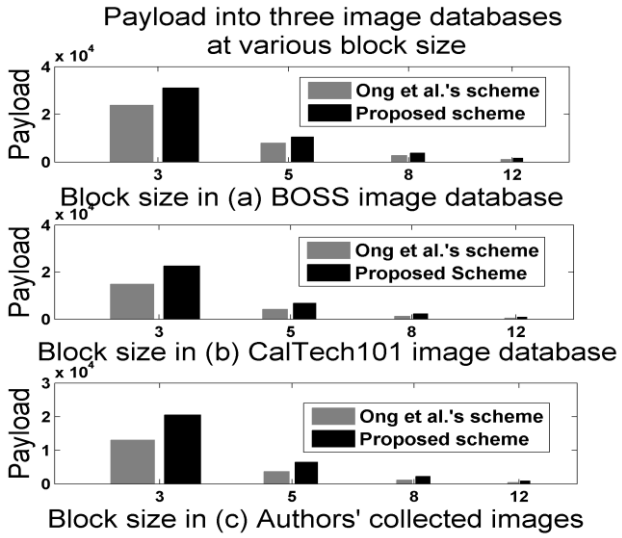


Figure 7.12: Average payload obtained into three image databases for different sizes of blocks.

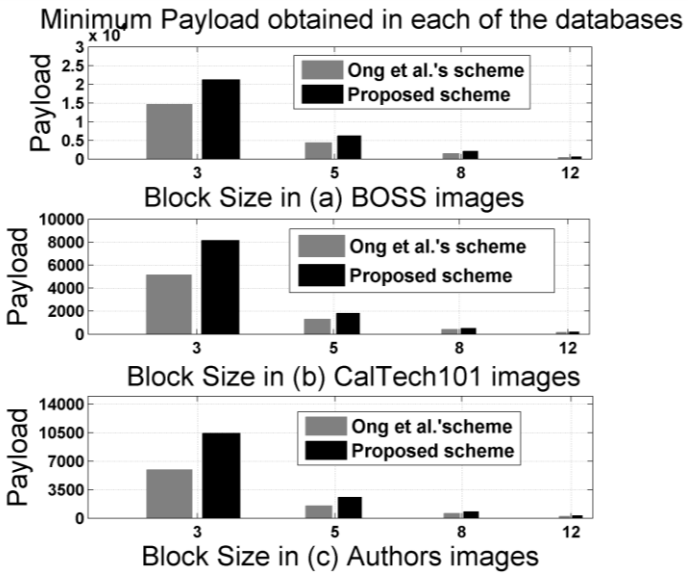
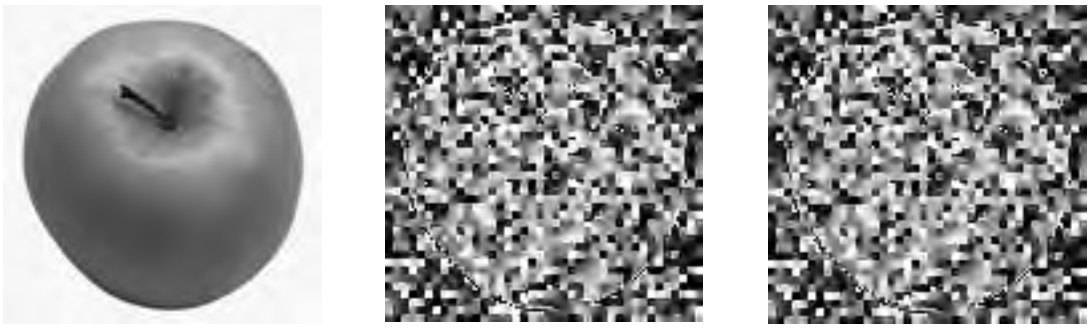


Figure 7.13: Minimum payloads obtained in each of the image datasets



(a) (b) (c)

Figure 7.14: Cover and stego images: (a) cover image; (b) stego image obtained by Ong *et al.*'s scheme; (c) stego image obtained by the proposed scheme.

7.5.3 Analysis of PSNR

The target of the proposed scheme is to destroy the image quality intentionally and notably. Figure 7.14 delineates that the cover image, here an apple, is destroyed noticeably by using both the Ong *et al.*'s scheme [71] and the proposed scheme. Nothing is recognizable about the cover information from these stego images. The figures state that visually the target of destroying the cover information in the stego image is achieved. To test the scheme statistically, the PSNR values of the resulted stego images are also measured.

Figure 7.15 and Figure 7.16 present the values of PSNR. The Figure 7.15 demonstrates the computed PSNR values for three different schemes in the first 100 images of the CalTech image dataset while the Figure 7.16 provides a comparison of PSNR values in three different image datasets for various sizes of image blocks. Figure 7.15 depicts that the Liao *et al.*'s scheme [52] presents the highest values for PSNR in all the images. The reason is that the scheme does not change the values of 50% contents of the embedding space. Further, the 3 LSBs of the rest 50% contents are flipped by the embedding procedure. This flipping operation ensures that difference between the cover and stego values are no more than 8. Consequently, compared with the embedding space, the mean square error (MSE) is no more than $8^2/2$, which implies that the PSNR values will never be less than 33dBm. However, the depicted PSNR values of the scheme are less than 33dBm in many images. This is because the scheme first encrypts the cover values and then implants bits by flipping the 50% of the encrypted values. On the other hand, the PSNR values in the Ong *et al.*'s scheme [71] and in the proposed scheme greatly depend on the pattern of the implanted bits. Additionally, the ranges of pixel values in the Ong *et al*.' s scheme and the range of absolute values of the prediction errors in the proposed scheme play an important role in destroying the image quality. For this reason, the PSNR values of these two schemes vary in the images in a wider range. Again, the flipped operation, introduced in the proposed scheme, allows more pixels to be changed from very large values to very small values. As a result, worse PSNR is presented by the proposed scheme.

In the Figure 7.16, it is examined for all categories of images that the PSNR values decrease for smaller sized image blocks, though the changes are happened nominally because the average ranging amount of pixel values in the smaller sized image blocks decreases. The number of gray partitions increases for the smaller valued range. Increased number of partitions enhances the probability of associating cover block with more number of partitions in the HAM policy and, thus, it, indeed, enlarges the length of the implanted message chunk. As the length of message chunk increases, the quantity of only ’0’ bit consisted message chunk decreases. If the chunk does not consist of only ‘0’s, the implantation of that chunk by the HAM policy confirms a certain change in the stego pixels.

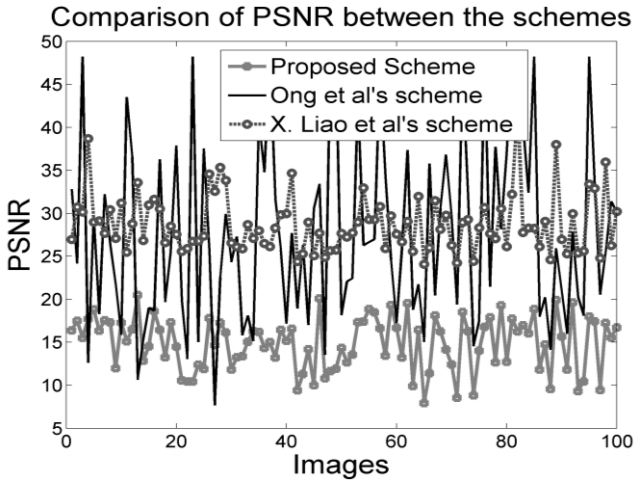


Figure 7.15: Comparison of PSNRs achieved in the first 100 images of the CalTech image dataset by different schemes.

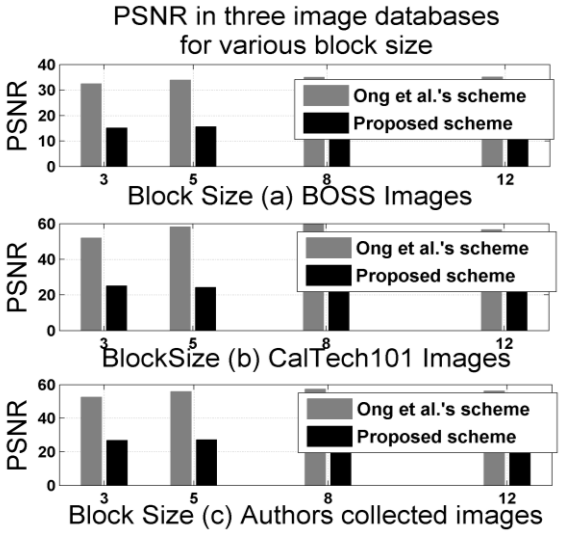


Figure 7.16: PSNR for different image datasets at different block sizes

PSNR in the proposed scheme is smaller than the same in Ong *et al*.'s scheme due to the following two reasons. Firstly, the ranges in the proposed scheme are smaller and thus, many blocks are shifted to distance gray parts. Secondly, the proposed scheme flipped a good number of stego pixels from very large values to very small values due to the use of Eq. (7.6).

Table 7.2: Detection of modified images in percentage by *gBL* method.

Image Database	Scheme	Detection rate (%) by *gBL* in Block Size of			
		3x3	5x5	8x8	12x12
BOSS	Ong *et al.*	72	67	72	76
	Proposed	71	71	68	69
CalTech	Ong *et al*.	86	77	94	97
	Proposed	86	81	95	98
Standard	Ong *et al*.	88	81	99	97
	Proposed	89	77	96	97

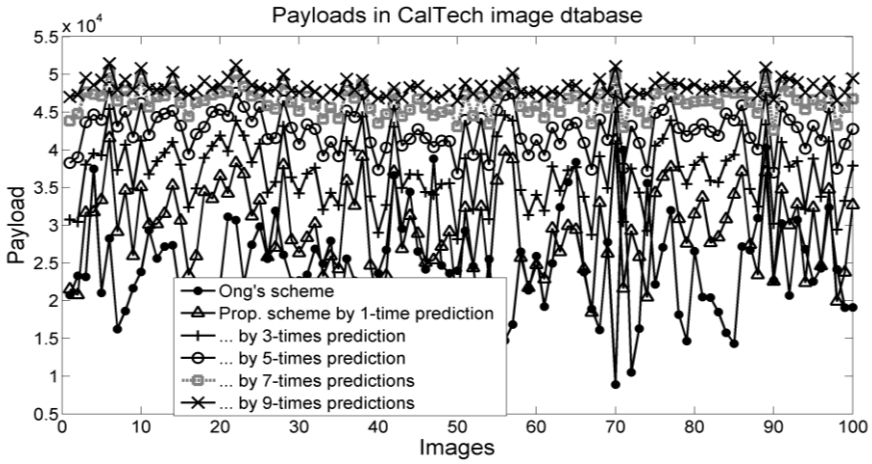
7.6 Resistance to Steganalysis

The steganalyzer measures in the stego image the possibility of the image being modified, but not the way of being modified. It also does not extract the stego contents. Hence, stego detection does not mean that the stego contents are detected. It is easy to detect the stego image of an intentional image distorted based scheme because the distances between the cover and the stego values are noticeable. The rate of stego detection depends on the rate of image distortions and it increases with the increment in the distortion rate. Consequently, the stego detection rate will be higher in both the proposed scheme and the Ong *et al.*'s scheme as these two schemes destroy the image intentionally. The steganalyzer is applied in this chapter just to check whether the steganalyzer provides higher detection rate because the higher detection rate ensures that the image is destroyed drastically. In this chapter, the generalized Benford Law (gBL) [23] is used to perform the steganalysis operation because it is easy to implement, faster in computing, good at stego detection and a very latest method. The detected results are listed in Table 7.2 for the proposed scheme and Ong *et al.*'s scheme. The values shown in that table are measured by experimenting the gBL stego detection method in the images of BOSS, CalTech and Standard dataset. The detection rates, that is obtained in all the sizes of block, by the proposed scheme in the CalTech images is a bit higher than the same in the Ong *et al.*'s scheme. In the other datasets, the matter of presenting the maximum detection rates by the gBL varies between the two experimented schemes for different size of the image block. Nevertheless, in all the stated results, the stego detection rates in both the Ong *et al*.'s scheme and in the proposed scheme are close to each other. For example, the detection rate varies in the Standard images by 1%, 5%, 3% and 0% for the block size of 3x3, 5x5, 8x8 and 12x12

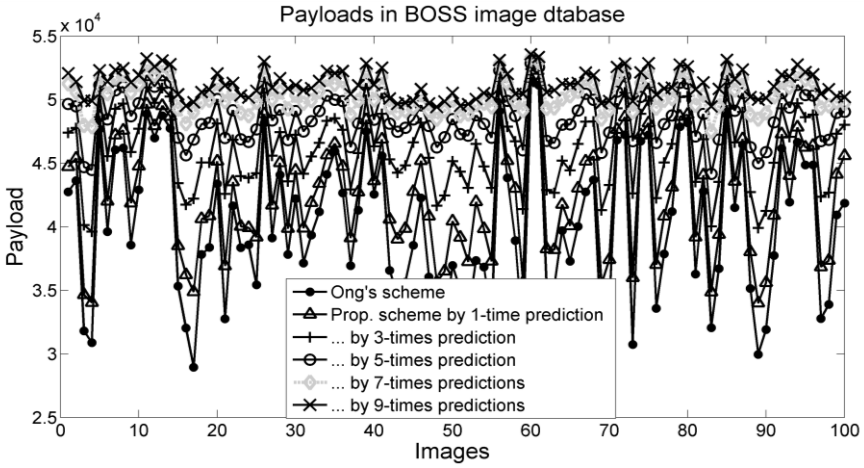
respectively. The minimum and the maximum stego detection rate that is computed in the proposed and the Ong *et al.*'s schemes are also analyzed. The minimum detection rates by the proposed and the competing scheme are {68, 67} in the BOSS, {81, 77} in the CalTech and {77, 81} in the Standard dataset and in the same manner, the maximum detection rates are {71, 76}, {95, 97} and {97, 99} respectively. These values state that the detection rates are very high and the detection rates in both the schemes are very close to each other in all the image datasets. The high values of detection rates confirm that the stego images are destroyed drastically.

7.7 Further Improvement of Embedding Capacity by Applying Repeated Prediction Process

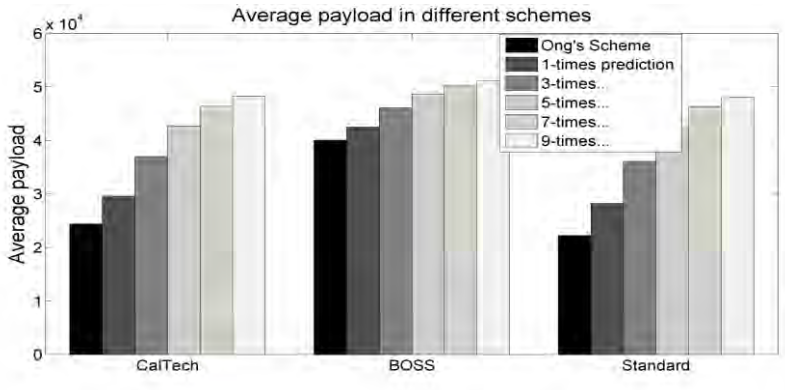
By the time, it is realized that the embedding payload increases for smaller valued ranges in a HAM scheme. In this section, repeated prediction policy is presented to reduce the magnitude of the range values in the prediction error space. In the first stage of the prediction process, the pixel values of a cover image are predicted. The absolute values of the prediction errors are measured as it was done in the single time prediction process as discussed in Section of 7.4.1. The absolute valued prediction errors are $E^{A}_{i,j}$. At the beginning of the second phase of the prediction process, the absolute errors $E^{A}_{i,j}$ are deemed as the pixel values of an image. The same predictor is applied to predict the values of $E^{A}_{i,j}$. After the prediction, the prediction errors and then the absolute values of the prediction errors are measured as like it was computed in the first phase of the prediction process. Considering these newly generated absolute valued prediction errors as another image, the whole prediction processes, i.e., from predicting values to measuring absolute prediction errors, is repeated. The scheme repeats the whole process for *n* times. Doing the repeated predictions, the scheme reduces the ranges of error blocks in its final level. Thereafter, the processes explained in Section 7.4.2 and Section 7.4.3 are executed to implant data bits.



(a)

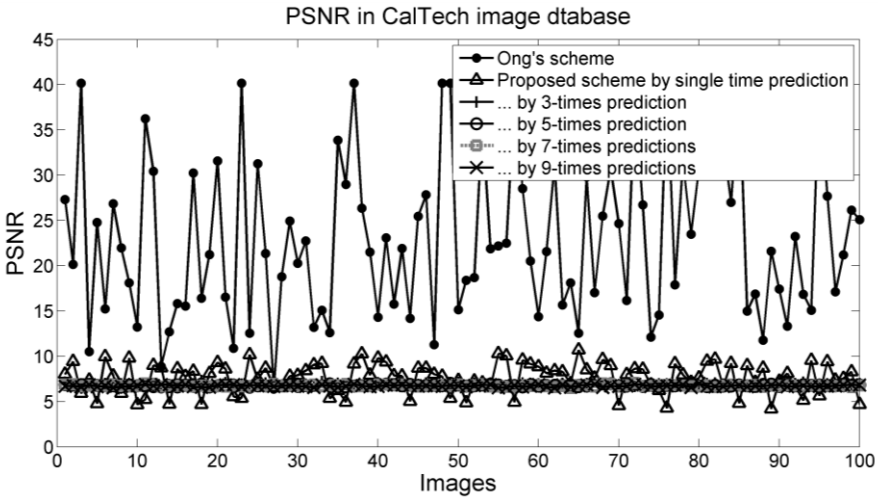


(b)

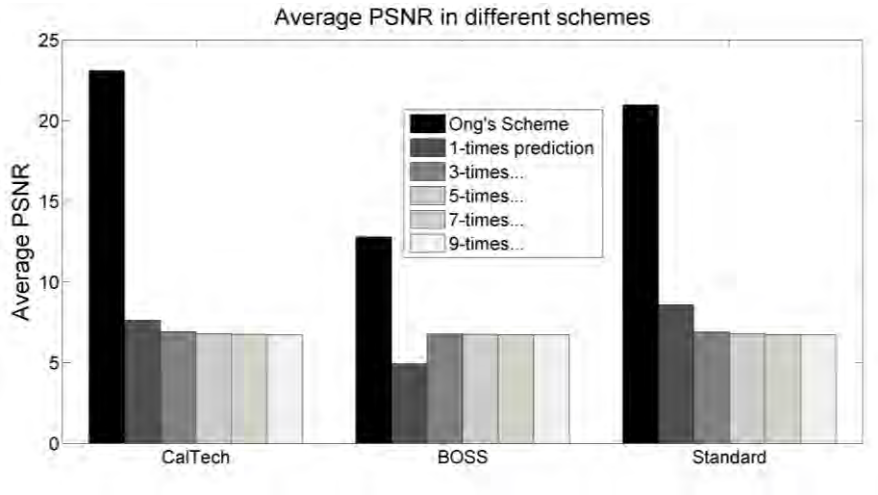


(c)

Figure 7.17Analysis on payloads in different image dataset: (a) the Caltech; (b) the BOSS; (c) the Standard image datasets



(a)



(b)

Figure 7.18: PSNR in different image dataset: in (a) the CalTech; (b) the Standard image datasets

7.8 Result Analysis and Discussion for RPBHAM Scheme

The RPBHAM scheme is experimented in different image datasets. The results on payloads and PSNRs are demonstrated in Figure 7.17 and Figure 7.18, respectively. The Figure 7.17(a) and Figure 7.17(b) present the payloads that are obtained by Ong *et al.*'s scheme and RPBHAM scheme for different values of n in the image datasets of CalTech and BOSS. In both the figures, this is noticeable that the payload increases with the increment of n, i.e., under n-times prediction policy. The results are demonstrated for 100 images. The average payloads for all the images in each image dataset are presented in Figure 7.17(c). This figure establishes the same fact that is observed in the other two figures. These three figures firmly establish the

claims. The PSNR values are also analyzed in Figure 7.18. Figure 7.18(a) states that the PSNR decreases when the value of n increases in the scheme of RPBHAM. The reason is explained in the Section 7.3.5. The PSNR in average is depicted in the Figure 7.18(b). This figure delineates that the average PSNR is very small in all the image dataset for the RPBHAM scheme. As the PSNR value is smaller in our proposed scheme, this can be concluded that the RPBHAM scheme deeply destroys the cover information and hence fulfills the objective.

7.9 Summary and Comments

Image quality degradation based embedding schemes are useful when the carrier itself is secret. There are lots of applications where the image is a secret and the other secret messages are embedded in that image before sending it to a destination. The stego image is destroyed up to a level such that the cover contents are not recognizable. As the scheme is free from managing the stego image quality, the methods can embed more data bits. Hence, the usability of such methodology is increasing day by day. In the first phase of this chapter, the provided solutions to the investigated limitations have improved the robustness of the existing scheme. In the second phase, the prediction error based HAM scheme has improved the embedding capacity notably. Finally, the repeated prediction error based scheme has boosted up the embedding capacity to another better level. The results are very promising. The proposed methods enhances the embedding capacity by several multiples of its competing schemes. The levels of degrading the image quality by the proposed schemes are also noticeable.

Chapter 8

Image Encryption Based Enhanced Embedding Scheme

In many clandestine applications, secret information, e.g., investigation reports, is implanted into an image document like forensic evidence before sending the document to the destination. In that case, both the contents of the documents and the implanted information are equally important and secret. To protect the document's own information, called the cover information, from being disclosed, many reversible data hiding (RDH) schemes first destroy the cover information intentionally and then embed message bits into these destroyed contents. A reversible process in the receiver end retrieves both the implanted message bits and the original image information. For the constraint of managing the reversibility, these schemes suffer from less embedding capacity, i.e., smaller ratio of embedded bits per pixel (bpp), because the reversible processes either are unable to implant bit(s) into every pixel or implant a chunk of message bit(s) into a group of pixels where the length of the message bits is smaller than the number of pixels in the group. This chapter proposes a novel distortion based RDH scheme that provides higher embedding capacity by implanting 2^n bits into every pixel, where $0 \leq n \leq 3$. In the proposed scheme, the documents are destroyed intentionally both before and after the data implantation task to strongly obliterate every information of the original image and the embedded bits. During this complete process, the scheme ensures seven levels of encapsulated securities and in consequence, strengthens the security of the mechanism. The maximum embedding capacity and the lowest level of image distortion are achievable up to 8 bpp and 5dB, respectively, in the proposed scheme. These two values are significantly dominating figures with respect to the same in its competing schemes.

8.1 Introduction

In the field of forensic, medical, military and satellite applications or in operational parts of the industrial control units, two parties communicate between themselves to exchange secret

information like personal information and images, military commands, medical images and medical history of patients, crime reports, investigation reports and forensic evidence. In this communication system, an application that uses image steganography implants the secret message bits into an official image like a person's photograph, forensic evidence, satellite image, medical image and scanned document. After the bit implantation task, that official image is termed as the stego image. The stego image is sent to the destination over a communication channel. The receiver applies the reversible process to extract the secrets as well as to reconstruct the original cover image. In that scenario, though the stego image provides the security to the implanted message bits, if the cover information remains visible in the stego image, any third party will grab the secrets of the cover. In such applications, the process of damaging the secret visual contents in the stego image is a good technique because then any third party will not be able to guess and retrieve the cover secrets from the transmitted stego. At the receiver end, the original cover images are reconstructed by a reversible mechanism. This is already stated in Chapter 7 that the embedding rules of [29, 42, 71] modify the cover values during the data implantation period. In these schemes, each block of pixels is equally translated by an amount in a direction. The translation amount depends on the pattern of implanted message chunk and range of block pixels. Still, that block shifting strategy cannot ensure pure distortions of the cover image because lots of the cover blocks are remained unchanged by conceiving the message chunk of 0 bits only. Many applications [52, 110], therefore, first destroy the cover image entirely by an encryption process and then embed the data into these destroyed values. These pre-distortion demanded embedding schemes suffer from less embedding capacity because, for the constraint of maintenance of reversibility, these schemes implants either a bit in a pixel [110] or a chunk of the message bits in a block of pixels [52], where the number of bits in the chunk is less than the number of pixels in the block. Additionally, many of these schemes [110] implant a large quantity of assistant information. Consequently, the pure embedding capacity is poor.

In this chapter, an intentional image destruction based reversible data hiding scheme is presented where encryptions are performed before and after the data embedment with two different 8-bit keys to maximize the level of distortions and the security of the implanted data bits. An algorithm is proposed to generate these two keys from another arbitrary 16-bit secret key, which is chosen by the data hider. This 16-bit key is placed in an arbitrary position in the image without destroying the values of two cover pixels. The 8-bit key generation algorithm increases the security of the system. The freedom of both choosing a 16-bit key of any value

by the data hider and placing the key at any position in the image have increased the robustness of the scheme. During the implantation, data bits are distributed over the binaries of each pixel in an equal distance to protect the least significant bit (LSB) extraction attacker from their successful mission. The number of implanted bits varies from pixel to pixel. This varying quantity of implanted bits in pixels has improved the security and the robustness of the proposed scheme. To further enhance the security, the dimension of the image is changed to make it more dissimilar to the cover image. In the proposed scheme, the embedding capacity is definable according to the demand of the system. The experimental results state that the proposed scheme dominates its competing methods [52, 53, 71, 110] by all of its measuring features like embedding capacity and image distortions.

The remaining parts of this chapter are organized into four sections. Section 8.2 illustrates the related works on which the proposed work builds its basement. The proposed scheme is detailed in Section 8.3. Section 8.4 delineates the performance of the scheme over the competing schemes. Finally, Section 8.5 concludes the chapter.

8.2 Related Schemes

As the proposed scheme intentionally destroys the quality of the cover image, two pre-distortion based schemes [52, 110], where the cover image is distorted first before starting the embedment process, and a benchmark scheme of distorting the image by data implantation rules [71], i.e., distorting during the implantation, is accounted as the related schemes. These schemes are described in Chapter 2. The proposed method of this chapter implants variant quantity of bits in the image pixels. To implant divergent quantity of bits in pixels, the concept of Liao *et al.* [53] is applied in this chapter. Though this scheme tries to manage better image quality, for the concept of implanting variant bits in the pixels, the scheme of [53] is also considered as a related work. The scheme proposed by Liao *et al.* [53] is briefly explained below.

The scheme in [53] works in the spatial domain. It implants bits by replacing the least significant bits (LSBs) of the image pixels. The scheme divided the cover image into blocks of four pixels. The average distance of the pixels from the minimum one in the block is measured. If the distance value is smaller than a threshold, the data bits to be implanted replaces a few numbers of LSBs; otherwise more LSBs are replaced by the similar number of message bits. This way, message bits are embedded in all the blocks by applying the rules of

LSB substitutions. The receiver end applies the reversible process to extract the exact quantity of implanted bits from each block by using the LSB extraction rules.

8.3 Proposed Image Encryption Based Embedding Scheme

The proposed scheme takes a cover image *I* of size $x \times y$. Each pixel at (*i, j*) location of the image is read by $I_{i,j}$. The proposed scheme produces an encrypted stego image *D* from *I* in three steps - first, the image *I* is encrypted using an 8-bits key *K*, say, the encrypted image is *C*; message bits are embedded into *C* which is then termed as the stego image *S*; the stego image *S* is further encrypted by another 8-bits key *R* to further strengthen the security of the system. The image generated by using the key *R,* say, the image *D* is called the encrypted stego image. These two encryption processes ensure the security of the implanted data as well as the contents of the cover media from being realized by an adversary. Thus, three levels of securities, e.g., two-time encryptions and a data concealment process are implemented. During the data implantation, the secret bits are distributed in an equal distance over the binaries of each of the contents of the image *C*. For this reason, either 1 bits, 2 bits, 4 bits or 8 bits are implanted in a content of *C* because each content of *C* is long of 8 bits and the 8 bits are equally dividable into either 1 bit, 2 bits, 4 bits or 8 bits. This bit distribution technique confirms the fourth level of the data embedment security as the LSBs attacking is not possible and the distance between two implanted bits is unknown to the third party. The number of implanted bits varies from pixel to pixel. A modified policy of [53] is used in selecting the number of bits for embedding into a pixel. Thus, a fifth-level of security is established in the proposed scheme. The stated keys *K* and *R* are generated from *L* and *M* respectively, where *L* and *M* are the parts of another encoder generated 16-bits secret key *E* such that $E = L \,||\, M$ where .||. stands for concatenation of binary string. The process of generating *K* and *R* from *L* and *M* promotes the safety of the system to the sixth level of security because other than knowing the exact process of generating *K* and *R* from *E*, the adversary cannot breed these two keys and thus, the challenger becomes fail to decrypt it. The process of generating *K* and *R* is explained in the Subsection 8.3.1. As a final step of the security measure, i.e. the seventh level of the system's security, the scheme generates hybrid stego image *H* by shuffling the values of the encrypted cover image *C* and the encrypted stego image *D* owing to make the *H* dimensionally and visually more dissimilar with the cover image *I*. All of these seven levels of

securities have ensured stronger protection of the implanted data in the proposed scheme against attacks.

8.3.1 Key Generation and Image Encryption

At the first stage of the proposed method, the data hider arbitrarily generates a 16-bit key E. The key E is either negotiated between the sender and the receiver or it is implanted into a specific part of the transmitted image. The E is partitioned into two disjoint parts L and M, i.e., $E = L \,||\, M$, as shown in Figure 8.1 based on the first two bits of E. The decimal value of the first two LSBs of E is used for pointing to one of the four predefined values as the length of L. Nevertheless, in the proposal, the L is allowed to be 2-bits, 4-bits, 8-bits or 12-bits. The L is formed by the defined number of most significant bits (MSBs) of E (i.e., $|L| \in$ {2-bits MSB, 4-bits MSB, 8-bits MSB, 12-bits MSB}, where |.| stands for the length of a string).

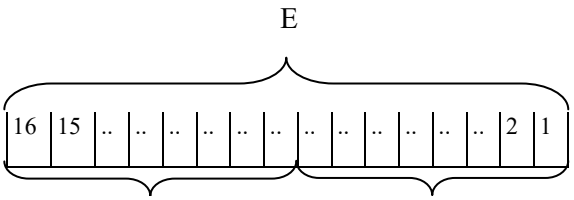


Figure 8.1: Encryption key and its sub-keys

```
If LSBs(E, 1, 2)='00'
        L= LSBs(E, 1, 2)
        M= LSBs(E, 3, 12)
        K=L||L
        R= LSBs(M, 1, 8) ⊗ LSBs(M, 9, 6)
If LSBs(E, 1, 2)='01'
        L= LSBs(E, 1, 4)
        M= LSBs(E, 5, 12)
        K=L||L
        R= LSBs(M, 1, 8) ⊗ LSBs(M, 9, 4)
Else if LSBs(E, 1, 2)='10'
        L= LSBs(E, 1, 8)
        M= LSBs(E, 9, 8)
        K=L
        R=M
Else if LSBs(E, 1, 2)='11'
        L= LSBs(E, 1, 12)
        M= LSBs(E, 13, 4)
        K= LSBs(L, 1, 8) ⊗ LSBs(L, 9, 4)
        R= M||M
End
```

Figure 8.2: Generation of encryption keys K and R.

The L and M are measured using the pseudo-code shown in Figure 8.2. In the code, the function LSBs(B, s, n) is used to extract n bits of information starting at the position s from a binary string B and the symbol $\otimes$ represents the bitwise exclusive-or operation. The Figure 8.2 also constructs two 8 bit keys K and R from L and M. These two keys K and R are used to

encrypt the working image at two different phases. The scheme first encrypts each pixel of *I* by the Eq. (8.1).

$$C_{i,j} = I_{i,j} \otimes R, \tag{8.1}$$

This encrypted image *C* is used in hiding data. The data embedment process is explained in the Subsection 8.3.2. After the data concealment, the image *C* is termed as the stego image *S*. The stego image *S* is further encrypted by another key *K* using the Eq. (8.2) so that any third party cannot extract the concealed message by using any steganalyzer.

$$D_{i,j} = S_{i,j} \otimes K, \tag{8.2}$$

```
If d ≥ 6 then
        n=8
Else if d ≥ 4 but d<6 then
        n=4
Else if d ≥ 2 but d<4 then
        n=2
Else
        n=1
End
```

Figure 8.3: Selecting the number of embeddable bits

8.3.2 Defining the Number of Implantable Bits in a Pixel

The proposed scheme implants a varying quantity of bits in the contents of the encrypted image *C*. The scheme distributes the bits to be implanted over the binaries of the encrypted pixels in *C*. To estimate the quantity of bits that could be embedded in (*i, j*) location of *C*, the scheme first measures the average value of a negotiated number, say *t*, of pixels within the contents of C that were accessed immediately before, e.g., the average of the encrypted pixels located from (*i, j-t*) to (*i, j-1*) in *C*. As the first *t* pixels do not have *t* number of immediate previous pixels, first *t* contents of *C* are not used for implanting bits. While working at (*i, j*) location of *C*, say, the average of *t* number of immediately accessed contents of *C* is *m*. The modulus value of *m* and 8, i.e., *d* = mod(*m*, 8), is computed. The value of *d* is the estimated quantity of bits that could be embeddable in the content, i.e., the pixel at (*i*, *j*) location. However, for the constraint of equally distributing the bits over the binaries of the contents of *C*, the scheme allows to implant in each content either 1 bit, 2 bits, 4 bits or 8 bits of information because each content of *C* consists of 8 bits and 8 bits of the contents are equally

dividable into the parts of either 1 bit, 2 bits, 4 bits or 8 bits. To distribute the n bits of information over the binary of a content of C in an equal distance manner, the exact value of n will be 1, 2, 4 or 8; rather than d. The value of n is measured by using the pseudo-code shown in Figure 8.3. While measuring n, the immediate previous t contents of C are accounted for improving the security only.

8.3.3 Data Embedment Process

Let the length of the message to be embedded be LL. The encoder implants LL bits of information in the encrypted image C. The implantation process is done taking into account that the receiver knows the value of E, t and LL. The bit length of E, t and LL are 16, 8 and 24, respectively. If the value of LL is not 24 bits, it is padded with a sufficient number of zeros to make it 24 bits long. These 48 bits are used as the assistant information. Without knowing the assistant information, the de-embedment is not possible. The 48 bits assistant information is negotiated between the communicating parties before sending the stego image. The assistant information is sent to the receiver end through another communication channel or implanting them at a separate location in the image C. If the assistance information is sent through embedding, the method of implanting them is made complex for the convenience of increasing the security of the system. The value of E is stored in C in an arbitrary position (u, v), i.e., E is stored in (u, v) and $(u, v+1)$. The value of u and v are stored at (1, 1) and (1, 2) of C, respectively. The value of t and LL are stored in the last four contents of the C. Let LL and E be divided into 8-bit components called L_1, L_2, L_3 and E_1, E_2, respectively. The implantation of assistant information is done using the pseudo-code containing eight assignment instructions as shown in Figure 8.4. In these expressions, both the keys K and R are used rather than a single one just to mislead the challenger during their accessing attempts.

$$C_{1,1} = u$$

$$C_{1,2} = v$$

$$C_{u,v} = E_1$$

$$C_{u,v+1} = E_2$$

$$C_{x,y-3} = t \otimes K$$

$$C_{x,y-2} = L_1 \otimes R$$

$$C_{x,y-1} = L_2 \otimes R$$

$$C_{x,y} = L_3 \otimes K$$

Figure 8.4: Implanting mechanism of assistant information

For these reasons, before starting the implantation process, the data hider picks eight contents, i.e., pixel values, of C from the locations of $(1, 1)$ and $(1, 2)$, two arbitrarily selected locations (u, v) and $(u, v + 1)$, and last four positions $(x, y - 3)$, $(x, y - 2)$, $(x, y - 1)$ and (x, y), i.e. the contents of $C_{1,1}$, $C_{1,2}$, $C_{u,v}$, $C_{u,v+1}$, $C_{x,y-3}$, $C_{x,y-2}$, $C_{x,y-1}$ and $C_{x,y}$ to protect them from being lost. These eight picked contents are concatenated with the secret message so that the decoder can reconstruct these picked values after the data extraction. Say, the concatenated result of the secret message and these eight content values is T. The length of T is, therefore, LL+64 bits. The resulting binaries of T are then implanted into the remaining contents of C.

During the implantation process of T, eight encrypted contents of $C_{1,1}$, $C_{1,2}$, $C_{u,v}$, $C_{u,v+1}$, $C_{x,y-3}$, $C_{x,y-2}$, $C_{x,y-1}$ and $C_{x,y}$ and t contents of $C_{1,3}$ to $C_{1,t+3-1}$ are not utilized in hiding data and thus, these are skipped by the data hider. These values assist the data extractor to retrieve the data and the cover values. The data hider implants each n bits of information, where $n \varepsilon \{1, 2, 4, 8\}$, into each remaining values of C. The value of n is computed by using the Figure 8.3. Let the n bits of data be $b_n...b_2b_1$, which is a part of T. During the implantation period, the embedded bits b_z, $1 \leq z \leq n$, are distributed over the binaries of the processed pixel in an equal distant manner, e.g., if n=4, the 1st, 3rd, 5th and 7th bits of each 8-bits encrypted pixel are modified, rather than into n LSBs. Such distribution not only enhances the security and robustness of the scheme but also increases the distortions in the stego image. Each processing pixel $C_{i,j}$ is first stored in a temporary variable P. After the data implantation, this stego pixel is assigned to the (i, j) location of the stego image S. Thus, C is kept unchanged. To realize the

implantation method, say, the binary of a working pixel $C_{i,j}$ is P, i.e. $P = Dec2Bin(C_{i,j})$ where Dec2Bin returns the binary of a decimal number. The scheme divides P into n components, each of which is l=8/n bits long, e.g. if $C_{i,j}$= 123, then P=01111011; and using Figure 8.3 the computed value of n is 4. The components of P are {01, 11, 10, 11}. Let each chunk of P is p_z, for $1 \le z \le n$, e.g., p_1=01, p_2=11, p_3=10 and p_4=11. The sequential concatenation of the entire p_z is P. Each b_z is embedded into each part p_z by using Eq. (8.3).

$$\tilde{p}_z = p_z \otimes b_z, \tag{8.3}$$

The stego pixel $\tilde{P}$ is formed by sequentially concatenating all the $\tilde{p}_z$ and $\tilde{P}$ is the $(i, j)^{th}$ pixel of the stego image S, i.e. $S_{i,j} = \tilde{P}$. By implanting all the message bits of T into the values of C, the stego image S is formed. The stego image is then again encrypted using the Eq. (8.2).

8.3.4 Hybrid Stego Image Generation

A hybrid stego image H is constructed by shuffling the pixels of the encrypted cover image C and the encrypted stego image D to change the dimension of the transmitted stego image by $2x \times y$ and to present a more meaningless image to the third party.

To shuffle the pixels of these two images, first, the hybrid image H of size $2x \times y$ is marked into black and white cells in a chessboard fashion. The pixels in C and D are assigned into black and white cells, respectively as it is shown in Figure 8.5. It is noticeable that the first row starts with C while the second row starts with D and like in the following. In this shuffling method, rather than starting by the pixels of C in every row, the chessboard-like distribution is chosen to influence the adversary to be strayed.

8.3.5 Message Extraction Process

The decoder receives the hybrid image H. It then writes all the black located pixel values into $C_{i,j}$ and all the white located pixel values into $D_{i,j}$ from H, where $1 \le i \le x$, $1 \le j \le y$. The values of $C_{1,1}$ and $C_{1,2}$ point to a position in C, where the key E is stored. Let $u = C_{1,1}$ and $v = C_{1,2}$. The bits of E were stored in $C_{u,v}$ and $C_{u,v+1}$ by the data hider. Therefore, the key E is extracted from $E = Dec2Bin(C_{u,v}) \,||\, Dec2Bin(C_{u,v+1})$. The pseudo-code shown in the Figure 8.2 is executed to generate the keys K and R from E. The number of associated contents t, which were used for computing the quantity of implanted bits is found by using the Eq. (8.4).

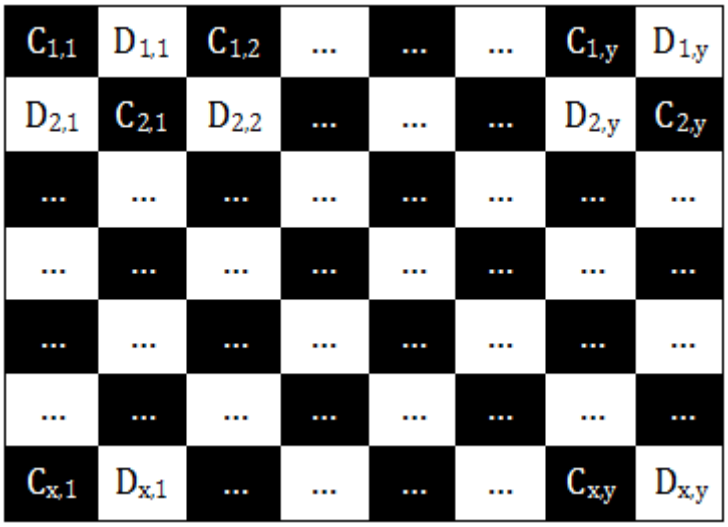


Figure 8.5: Formation of hybrid stego image *H*.

$$t = C_{x,y-3} \otimes K, \tag{8.4}$$

Similarly, the last three pixels of *C* decrypted using the Eq. (8.5) are used to find out the length of total embedded bits *LL*.

$$\left.\begin{aligned} L_1 &= C_{x,y-2} \otimes R \\ L_2 &= C_{x,y-1} \otimes R \\ L_3 &= C_{x,y} \otimes K \end{aligned}\right\} \tag{8.5}$$

where, $LL = L_1 \,||\, L_2 \,||\, L_3$. The Eq. (8.6) decrypts the encrypted stego image *D* by the decryption key *K* to find the stego image *S*.

$$S_{i,j} = D_{i,j} \otimes K \tag{8.6}$$

Let $\tilde{P}$=*Dec2Bin*($S_{i,j}$), $Q = Dec2Bin(C_{i,j})$ and $\tilde{Q} = \tilde{P} \otimes Q$. One bit of data was embedded into every chunk of *l* bits of $\tilde{P}$, where $l = {}^{8}/_{n}$. The value of *n* is measured for each pixel of C from the immediately accessed *t* pixels. The process is stated in the Section 8.3.2. Every $(z-1)\times l+1$ positioned bit of $\tilde{Q}$, where $1 \le z \le n$, represents the bit of message stream that was implanted by the encoder end at (*i, j*) location. The process of extracting message bits from each (*i, j*) location of stego image *S* is outlined in Figure 8.6.

The Figure 8.6 does change the stego pixel at (*i, j*) locations, where $(i,j) \in \{(1,1)$ to $(1,t),\ (u,v),\ (u,v+1),\ (x,y-3),\ (x,y-2),\ (x,y-1),\ (x,y)\}$. The extracted messages from all the stego pixels are concatenated to form the final message string. The

extracted message also contains the original values of $C_{1,1}$, $C_{1,2}$, $C_{u,v}$, $C_{u,v+1}$, $C_{x,y-3}$, $C_{x,y-2}$, $C_{x,y-1}$ and $C_{x,y}$. These values are reconstructed to form the original encrypted image C. Finally, the cover image I is generated by using the Eq. (8.7).

$$I_{i,j} = C_{i,j} \otimes R \tag{8.7}$$

1. $\tilde{P} = Dec2Bin(S_{i,j})$
2. $Q = Dec2Bin(C_{i,j})$
3. $\tilde{Q} = \tilde{P} \otimes Q$
4. $b_z = \tilde{Q}_{(z-1)\times l+l}$, for $1 \le z \le n$

Figure 8.6: Extracting message from a single stego **pixel**

8.4 Analysis of the Experimental Results

The experiments are conducted on different image datasets in MATLAB. A few of these are presented in Figure 8.7. The objective of the research work is to increase both the embedding capacity and the image distortions. Therefore, these two are analyzed in this part of the thesis. The proposed method is compared to the scheme proposed by Ong *et al.* [71], Liao *et al.'s* [52], Zhang *et al.* [110] and Liao *et al.'s* [53]. Though the scheme presented in [53] is not an intentional image distortion based process, this scheme is used in the comparison as the concept of variant bit implantation of this scheme is used in the proposed work. Very generally, the scheme in [53] provides higher image quality than the other experimented schemes.

8.4.1 Capacity Analysis

In two different experiments, the proposed scheme is allowed to implant up to 8 bpp and the variable amount of quantity as required. The results of 50 images are depicted in Figure 8.8. The experimental results delineate that the proposed scheme dominates the others noticeably. While the competing schemes presented in [52, 71, 110] fail to embed even 1 bpp, the proposed scheme implants up to 8 bpp. The scheme presented in [53] provides an embedding capacity within 2bpp to 3bpp because according to the objective of the scheme, to preserve better image quality, it is allowed to implant either 3 bits or 2 bits in each pixel. The embedding capacity of 8bpp is achieved by fixing n=8. This capacity is several multiples of

that found in the other schemes. In another experiment, the value of n is predicted from 4 previous pixels for t=4. The achieved capacity varies from 2.1 bpp to 5.7 bpp depending on the image properties. In 92% of images (46 out of 50), the obtained embedding capacity in the proposed scheme is higher than the highest performing competing scheme [53]. Thus, this is proved that the proposed scheme outperforms in all the experimented images and the obtained capacity is several multiples of others.

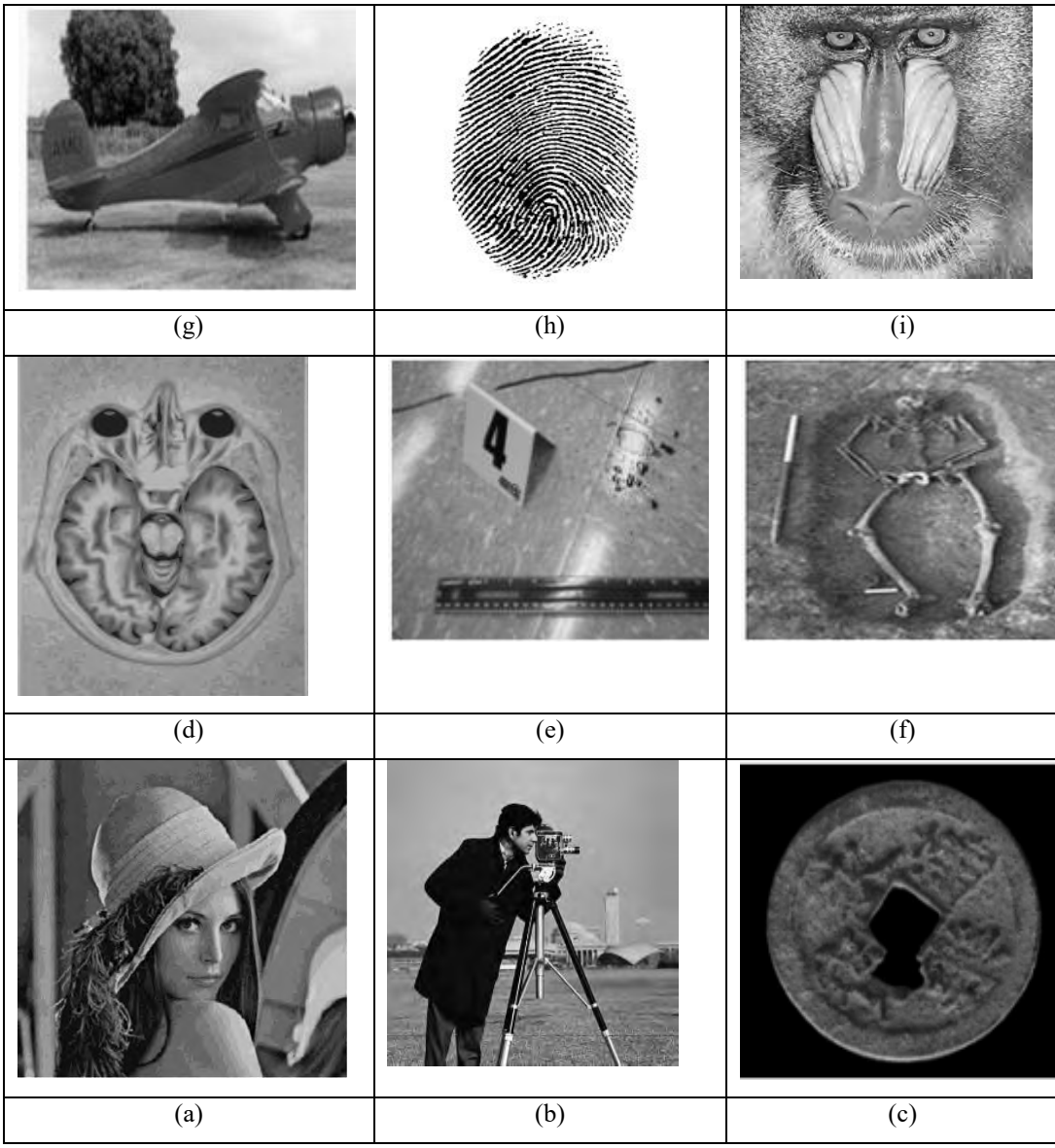


Figure 8.7: Sample cover images: (a) Lena; (b) Cameraman; (c) Old China coin; (d) X-Ray; (e) Crime zone; (f) Burial; (g) Plane; (h) Fingerprint; and (i) Mandril.

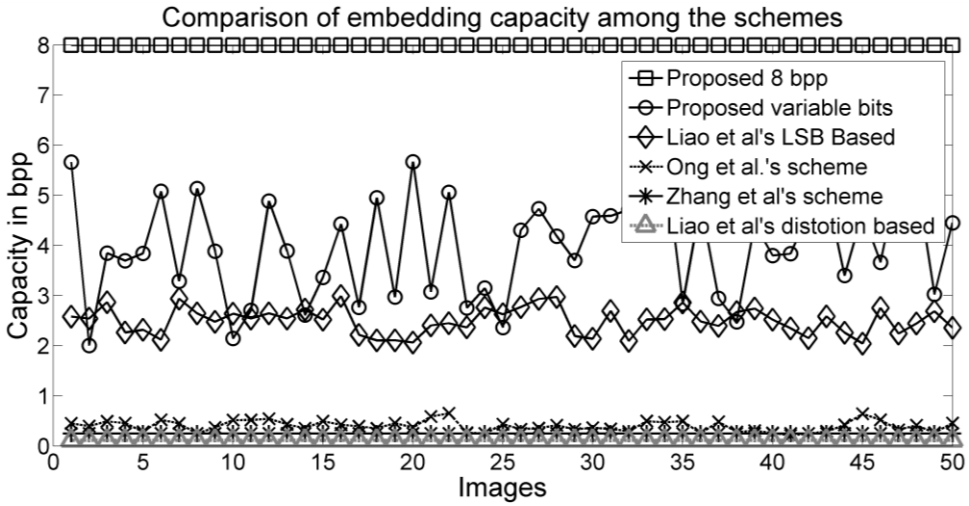


Figure 8.8: Capacity of the proposed and its competing schemes.

The embedding capacity, presented for the proposed scheme, is measured regarding the image size of $x \times y$. Nevertheless, the size of the hybrid image is $2x \times y$. The hybrid image is transmitted over the Internet. Hence, with respect to the size of the hybrid image, the embedding capacity of the proposed scheme is half of the presented figure. Hence the embedding capacity is 4 bpp for *n*=8 and 1.1 bpp to 2.9 bpp for variant bits implantation. These figures also noticeably dominate the achieved embedding capacity found by the schemes presented in [52, 71, 110] for all the images and [53] for many images.

8.4.2 Distortion Analysis

The cover image, the encrypted cover image, the encrypted stego image and the hybrid stego image of the fingerprint image are demonstrated in Figure 8.9(a), Figure 8.9(b), Figure 8.9(c) and Figure 8.9(d), respectively. From the encrypted cover image shown in Figure 8.9(b) and the encrypted stego image shown in Figure 8.9(c), nothing is realizable about the cover information; because the stated process of the proposed scheme destroys the visual and statistical information about the cover image. The hybrid stego image is depicted in Figure 8.9(d). This is also like a noisy image. Hence, the objective of destroying the cover information in the stego image is fully achieved through the proposed scheme.

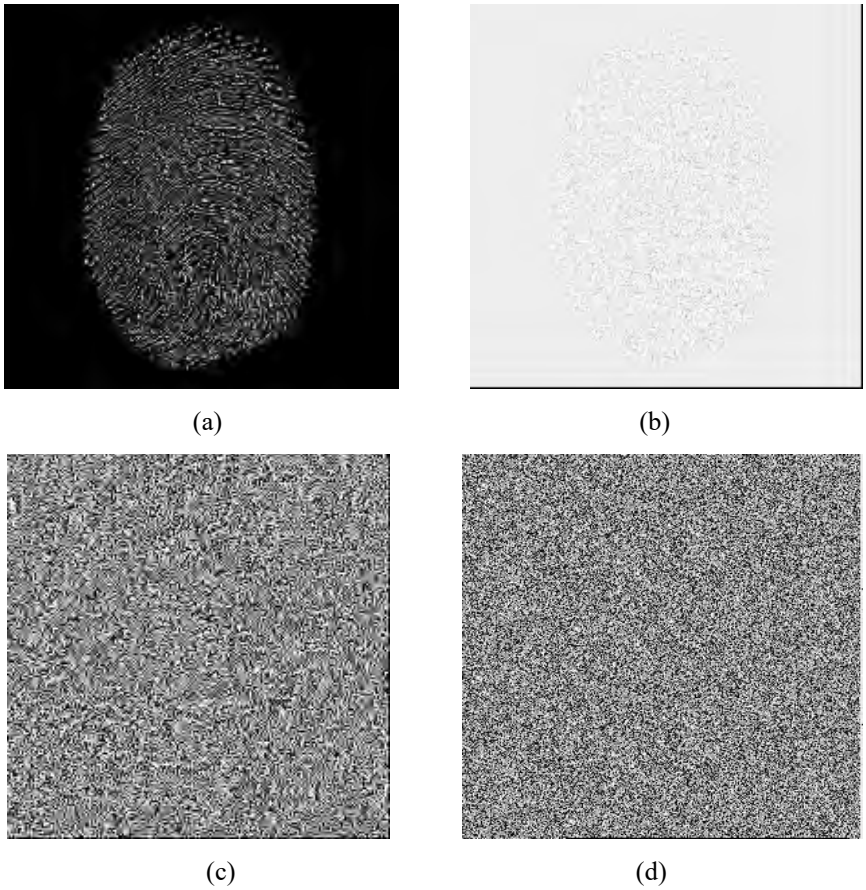
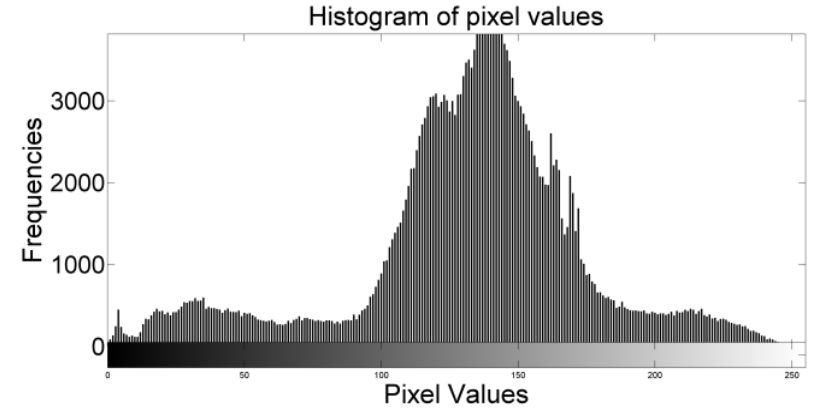


Figure 8.9: Images at various processing steps: (a) Fingerprint; (b) Encrypted image; (c) Stego image; and (d) Hybrid image

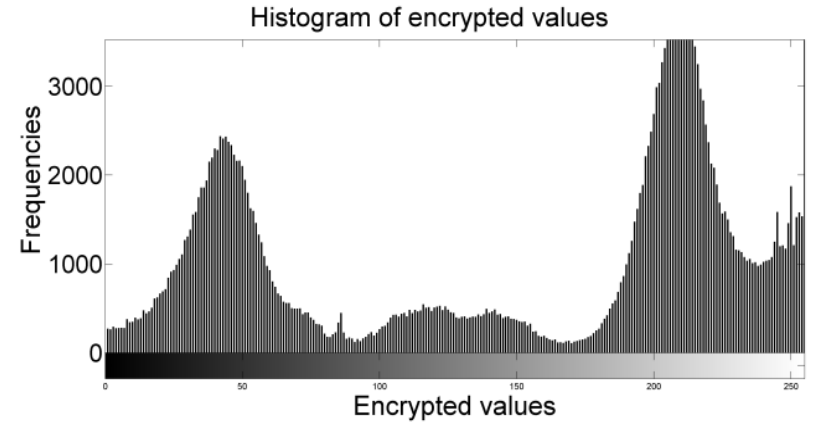
The distortions in the stego image and in the encrypted image are found due to the abrupt changes to the image pixels. Therefore, the pixel histograms of the cover image, encrypted cover image and encrypted stego image exhibit dissimilar properties, as it is shown in Figure 8.10(a), Figure 8.10(b) and Figure 8.10(c), respectively.

To compare the distortion levels, the quality of the stego image is measured by the peak signal to noise ratio (PSNR). The Eq. (8.8) is used to measure the PSNR values.

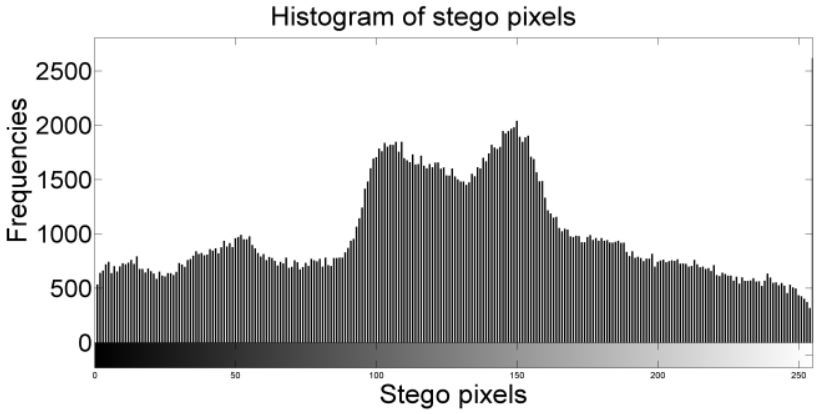
$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right) \tag{8.8}$$



(a)



(b)



(c)

Figure 8.10: Histograms of (a) the cover image; (b) the encrypted image; and (c) the stego image, respectively.

The mean square error (MSE) is computed by using the Eq. (8.9).

$$MSE = \frac{1}{x \times y} \sum_{i=1}^{x} \sum_{j=1}^{y} \left(S_{i,j} - I_{i,j} \right)^2 \tag{8.9}$$

The results are depicted in the Figure 8.11. The results state that the proposed scheme provides the lowest PSNR values in all the images because the proposed scheme encrypts the image for two times and implants message bits at different but equally apart positions in the binaries of each content. The distortion level achieved in [53] is not demonstrated because the objective of this scheme is to manage higher image quality while the proposed scheme tries to maximize the distortion level.

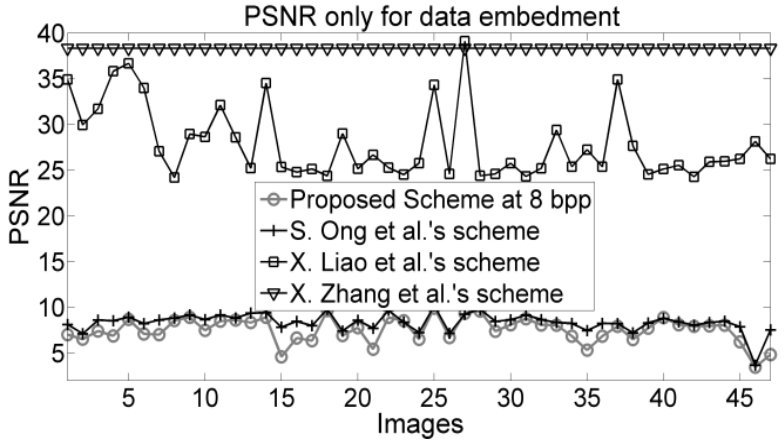


Figure 8.11: Comparison of PSNR values among the schemes

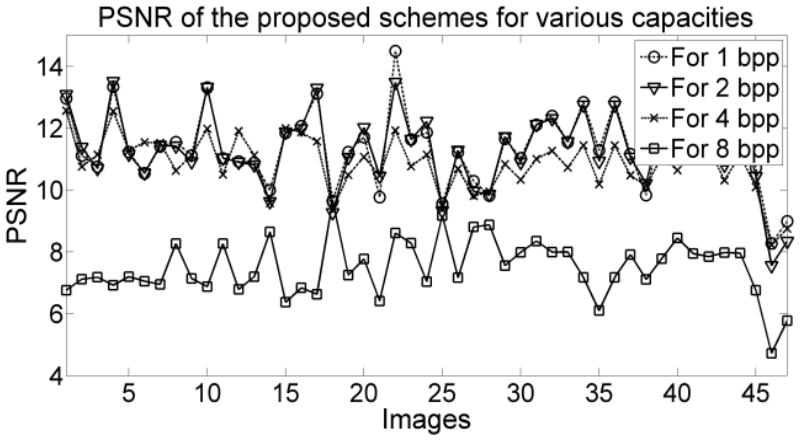


Figure 8.12: PSNR comparisons for various embedding capacity

To check the state of distortions for various levels of embedding capacities, the proposed scheme embeds into each cover image separately for four times by fixing n=1, n=2, n=3 and n=4 respectively. First, 1 bit is embedded into each pixel and the level of distortion is measured. Thereafter, 2, 4, and 8 bits are implanted in each pixel in subsequent executions and the distortion amounts are measured respectively. The distortions for the embedding capacity of 1 bpp, 2 bpp, 4 bpp and 8 bpp are analyzed in the experiments. The results are demonstrated in the Figure 8.12. It is observed that the values of PSNR decrease in all the images when the embedding capacity is increased. This is happening because, with the increment in the embedding capacity, more bits of each pixel are altered in the bit implantation phase. This, indeed, increases the number of changes in the stego contents regarding their cover contents. As a result, the PSNRs decrease in the proposed scheme and that decrease is proportionate to the embedding capacity.

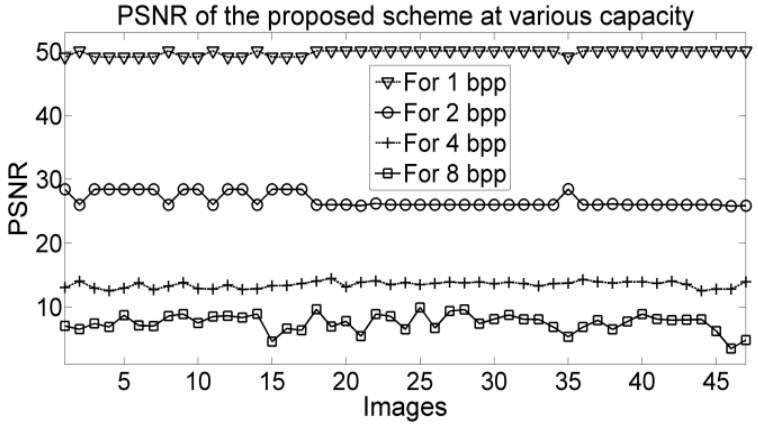


Figure 8.13: PSNR of the proposed scheme for only data embedment at various capacities

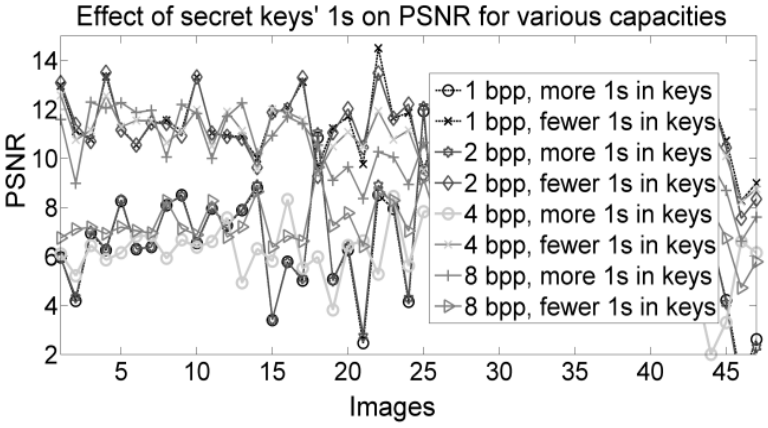


Figure 8.14: Effect of encryption keys to PSNR values

Both the encryption and the data concealment processes are responsible for lowering the values of PSNR. The effects of only the data embedment on the values of PSNR, excluding the effect of the image encryption, for different levels of embedding capacity are also examined in the proposed scheme. The results are delineated in Figure 8.13. The PSNR is decreased with the increment in the embedding capacity. The reason is same as it is observed in the Figure 8.12. When the embedding capacity is improved, the scheme implants more bits per pixel. This implies that the scheme alters more numbers of bits in each pixel's binary.

8.4.3 Effects of Encryption Keys on Image Distortions

Encryption keys do not have any effect on the embedding capacity; however, these affect the image distortions by the frequency of '1's in their binary values. It is investigated that keys with more '1's in their binary values destroy the image on a large scale because during the exclusive or operation in the Eq. (8.1) and the Eq. (8.2) each binary bit 1 in the encryption key modifies the respective binary bit in the image pixel. To check the effect, the values of key {*R, K*} are set to {100, 137} and {235, 183} in two separate execution periods. The produced results are compared. The key values of {235, 183} contain more '1's in their binaries than for the key values of {100, 137}. The results are demonstrated in the Figure 8.14. In the figure, it is noticeable that the number of '1's in the binaries of the keys plays an important role in affecting the values of PSNR. For all the investigated embedding capacities, it is observed that the values of PSNR decrease for the uses of keys with more '1's. Thus, it is found that {235, 183} valued keys destroy the image quality more than the keys of {100, 137}.

8.5 Summary and Comments

The distortion based reversible data hiding schemes do not care about the quality of the image. Rather, these try to destroy all the cover information in the stego image including the smallest one. These schemes provide higher embedding capacity because it is easy to manage the pixel values during the implantation of more quantity of message bits in a distorted image. Nevertheless, to the best of the author's knowledge, none of the reversible schemes in the literature provide an embedding capacity of 8 bpp. The proposed work shows enough novelty in achieving both the embedding capacity of up to 8 bpp and degrades the image quality up to a value of 5dB. Besides, it offers the data hider either to embed different quantity of bits into the pixels or to select the embedding capacity to one of the four values – 1 bpp, 2 bpp, 4 bpp

and 8 bpp. The proposed scheme is very effective in hiding the large volume of data and securing the secret messages and evidence related to forensic, medical, military, law-enforcing agency application. The seven levels of security features are implemented into the scheme in an encapsulation way. Hence, breaking the security is difficult. As a whole, it will be a useful contribution to the field of reversible data hiding arena.

Chapter 9

A Comparative Study of the Proposed Schemes

The thesis presents a good number of proposals to improve the embedding performance of predictive reversible image steganographic schemes. The thesis enhances the embedding capacity and the image quality by improving the prediction accuracy of multiblock center reference based predictor in Chapter 3. The chapter also narrates the ways of implanting bits for multiple times into the errors of a single layer in the prediction error histogram. Chapter 4 employs multiple predictors to compute optimal prediction errors owing to enhance the number of embeddable errors. A method of efficiently utilizing the embeddable errors in multi-layer approach is presented in Chapter 5. The bit implantation process, in Chapter 6, employs two different local pattern codes for the first time in this field to increase the embedding capacity and the quality of the stego image. Chapter 7 and Chapter 8 present two different image distortion based techniques where the embedding performance is improved notably by different mechanisms. Each of these proposed schemes has its own objectives and usefulness in a specific scenario. Therefore, it is hard to compare all these schemes by a common property. In this chapter, these schemes are grouped into several categories based on their relational properties and then performance comparisons are performed among the schemes in a group.

9.1 Introduction

The reversible image steganographic schemes either take care to preserve the image quality while data implantation, e.g., schemes of Chapter 3 to Chapter 6, or destroy the image quality intentionally by the embedding process, e.g., Chapter 7 to Chapter 8. The predictive reversible data hiding schemes employ one or more predictors to either improve the embeddable contents in the embedding space or estimate the number of bits that are allowed to be embedded in a content(s) by the applied scheme. In the proposed scheme of Chapter 3 and Chapter 4, the embeddable quantity is improved by improving the prediction accuracy. The embeddable

contents are efficiently used for multiple times data embedment in a part of Chapter 3 and Chapter 5. The objective of using local binary pattern code (LBP), in Chapter 6, is to enhance the embedding capacity of the single layer (SL) data embedment process at its highest level. The same chapter uses local ternary pattern code (LTP) to increase the quality of the stego image when the demanded embedding capacity is small. Chapter 7 and Chapter 8 destroy the cover information in the carrier image either by embedding rules or applying an encryption method during and before the data implantation, respectively. Both the schemes estimate the number of bits that is embeddable in a block or in a pixel. The scheme in Chapter 7 applies a predictor at its pre-processing stage so that the data implantation phase can estimate a higher value as a number of bits that is embeddable in a block. Thus, it is found that each scheme has its own objectives and it is hard to make a general comparison among the schemes. Rather, the findings state that the schemes relate themselves by their attempts of either (i) improving the prediction accuracy presented in Chapter 3 and Chapter 4, (ii) embedding multiple times as presented in Chapter 3 and Chapter 5, (iii) embedding into single layer for single cycle times (in Chapter 3, Chapter 4 and Chapter 6) and (iv) destroying image quality (in Chapter 7 and Chapter 8). In this chapter, the proposed schemes are compared among themselves based on these stated five relations. In the experiment, ten images are, first, randomly selected from image datasets, stated in Section 2.8. The experimental results conducted on these ten images are demonstrated in this chapter.

The rest of the chapter is organized into two more sections. Section 9.2 delineates the comparison results of the proposed schemes from different perspectives. Last Section 9.3 concludes the chapter.

9.2 Comparing the Proposed Schemes at Different Perspectives

Five different types of relational properties are established in section 9.1 to make comparisons among the schemes in each relational group. Based on these relational properties, the schemes are compared in the following four subsections.

9.2.1 The Schemes that Work to Improve the Prediction Accuracy

Most of the prediction error based embedding schemes mainly concentrate on improving the prediction accuracy during their workout for the development of embedding space because the

improved predictor presents more embeddable errors. The single layer data embedment process based on the improved predictor is presented in Chapter 3 and Chapter 4. Comparison of payloads and PSNR are presented in the following Figure 9.1 and Figure 9.2 for these two schemes.

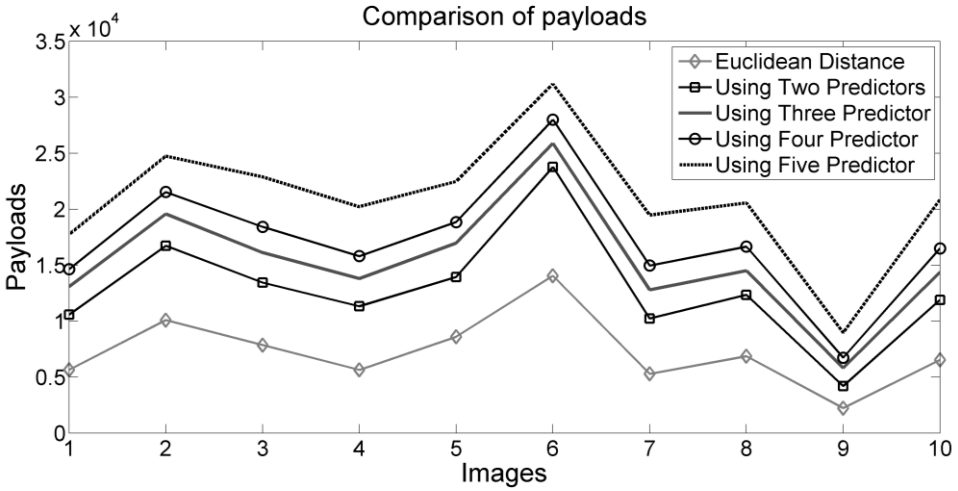


Figure 9.1: Comparison of the payloads of the prediction accuracy improvement based schemes

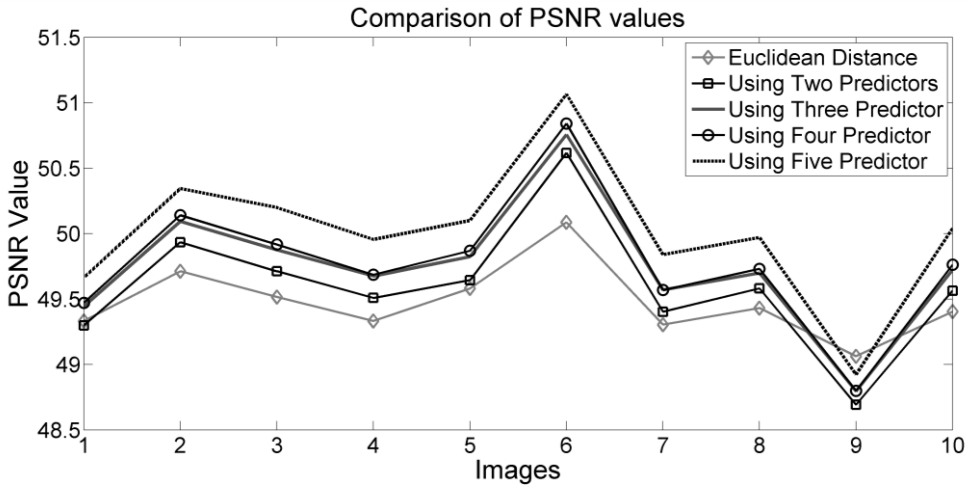


Figure 9.2: Comparison of the PSNR values of the prediction accuracy improvement based schemes

Figure 9.1 demonstrates that the multi predictor (two, three, four and five predictors) based schemes, proposed in Chapter 4, provide higher embedding payloads than the single layer Euclidean distance based scheme presented in Chapter 3, because with the application of more number of predictors, the scheme presented in Chapter 4 yields more prediction accuracy. This means that the scheme generates a good number of embeddable errors that is

larger than the one yielded with a single predictor. The same scenario is investigated for PSNR value in the Figure 9.2. It has already been stated in Chapter 3 and Chapter 4 that the more the embeddable errors, the less the image distortions. The number of embeddable errors increases with the application of more number of predictors. That is why the PSNR values of multiple predictor based schemes are higher.

Though the multi predictor based schemes provide better payloads and PSNR values than the single predictor based schemes, the processing time of the multi predictor based schemes is higher. That processing time increases with the increment of the applied predictors, as shown in Table 9.1. The time complexity is not a notable issue if the scheme implants secrets into a single image. However, it will be accountable if the scheme requires implantation of a large secret stream into more than a single image.

Table 9.1: Execution times for the different schemes in seconds.

Schemes	Image									
	1	2	3	4	5	6	7	8	9	10
Euclidean	7.5	7.7	7	6.2	7.9	5.6	5.3	5.6	7.1	6.8
2-Preictor	8.8	6.07	7.03	6.01	7.3	5.6	5.5	6.1	7.9	6.6
3-Predictor	9.4	7	7.8	7.4	7.9	6.2	6.2	6.6	8.5	7.2
4-Predictor	17.5	13.4	15.1	12.4	15.3	11.7	12.1	12.9	16.9	14.7
5-Predictor	29.7	21.6	23.4	20.4	24.6	19.4	19.2	20.4	26.7	23.5

9.2.2 The Schemes that Implant for Multiple Times

The multi-time data embedment process implants message bits in the embeddable errors for multiple times. The objectives of embedding multiple times into only a few high-frequency errors rather than into more errors for single time are explained in Chapter 3 and Chapter 5. The scheme in the Chapter 3 implants into the errors of a single layer while the scheme in the Chapter 5 does the same into the errors of more than one layer. When the size of the secret message is not too big, but errors of a single layer are unable to accept the secrets by a single time data embedment process, the multiple time embedment process is then employed. On the other hand, if the size of the to-be-embedded data is large, the multilayer, multi-time embedment process is applied because the later scheme provides higher image quality.

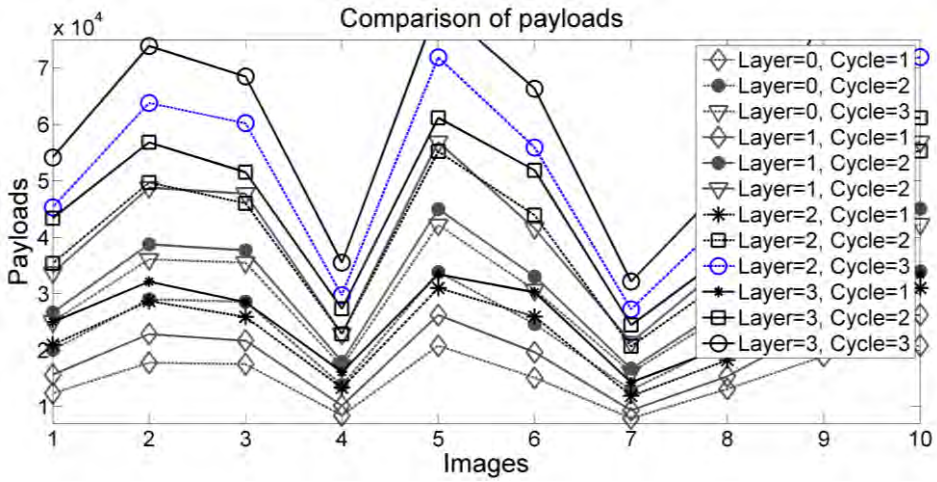


Figure 9.3: Comparison of payloads among the single layer, the multilayer and the multi-cycle schemes

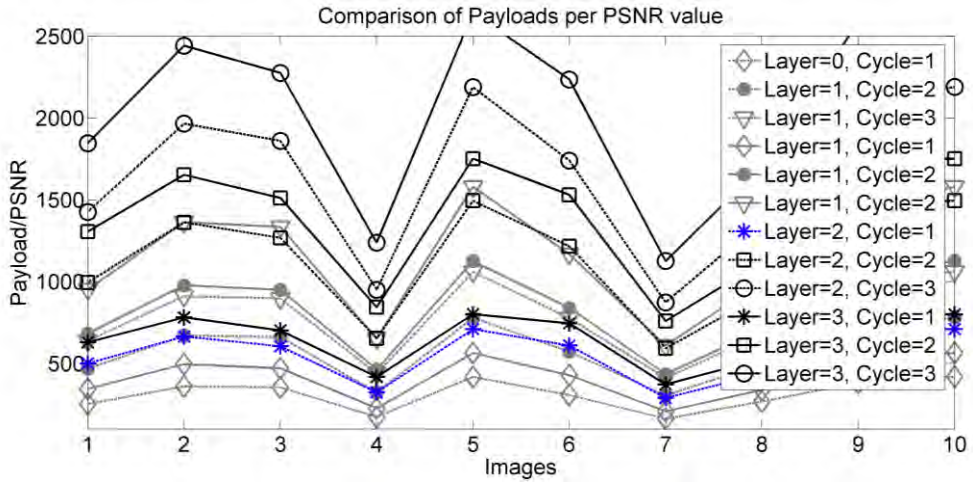


Figure 9.4: Achieved payloads per PSNR in single layer, multilayer and their multi cycle schemes

The results in Figure 9.3 states that the multi-time data embedment schemes provide higher embedding payloads. In the figure, layer 0 means single layer data embedment while the layers, greater than 0, stand for multilayer data embedment processes. In the figure, the results up to three times data embedment are demonstrated. The payload increases for both the increment of layer values and embedding cycles. The finding also states that the payloads of multilayer, multi cycle schemes are greater than the payloads of a single layer, multi cycle schemes. Figure 9.4 also delineates the same scenario where the payloads per PSNR value increases for both increasing the embedding cycle and in multilayer schemes.

9.2.3 The Schemes that Implant into Single Layer for Single Time

In this research work, Chapter 3, Chapter 4 and Chapter 6 present the process of implanting bits into the errors of a single layer, i.e., in the error values of -1 and 0, for a single time. The scheme in the Chapter 3 improves the prediction process of [32] and thus enhance the embedding capacity. The scheme presented in Chapter 4 presents a novel work that improves the prediction accuracy by employing multiple predictors in its prediction phase. Chapter 6 proposes two other novel works in which the first one achieves an embedding capacity of 1bpp and the second one improves the image quality when the requirement of embedding capacity is very small. A comparison of embedding payloads and PSNR values of these schemes are presented in Figure 9.5 and Figure 9.6, respectively.

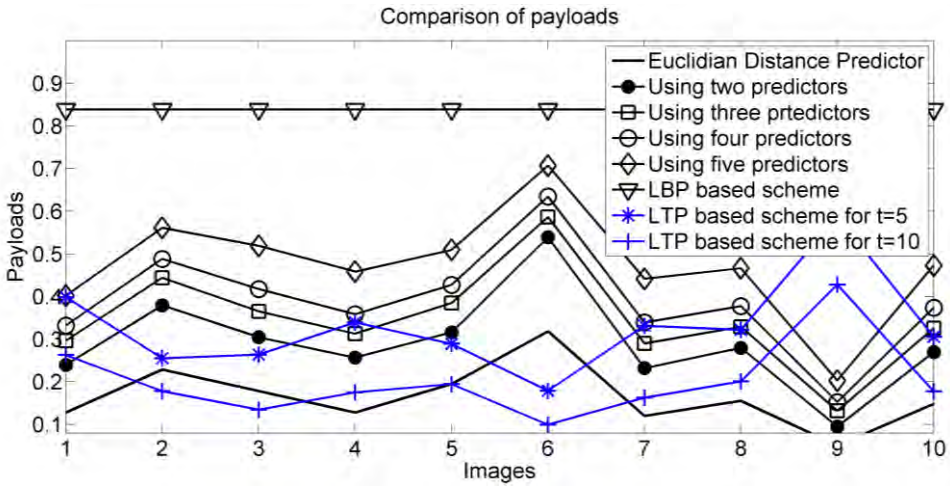


Figure 9.5: Comparison of payloads between the single layer and the single cycle schemes.

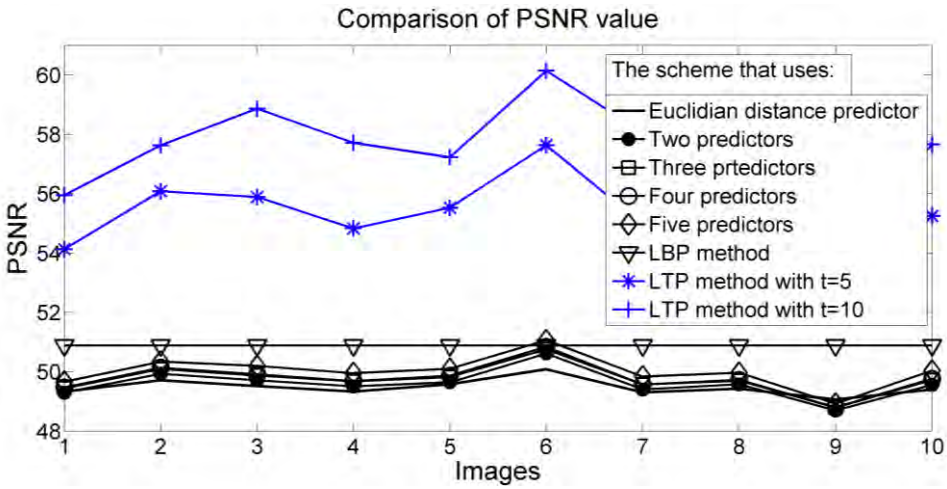


Figure 9.6: Comparison of PSNR values between the single layer and the single cycle schemes

Figure 9.5 demonstrates that the LBP based scheme provides highest embedding capacity and it is about 0.83bpp. The next highest schemes are the multi-predictor based processes. Therefore, these two processes are recommended for meeting large embedding capacity. Figure 9.6 delineates that the LTP based schemes provide noticeable larger PSNR value. When the requirement of embedding capacity is small, the LTP based scheme provides better performance. The LBP based scheme also demonstrates better image quality. For large size of to-be-embedded data, the LBP based scheme will give the highest embedding performance. Nevertheless, the LBP based scheme sends assistant information to the destination through another communication channel. If it is considered as a drawback of the system, the uses of the multi-predictor based schemes in implanting large data could be a better solution.

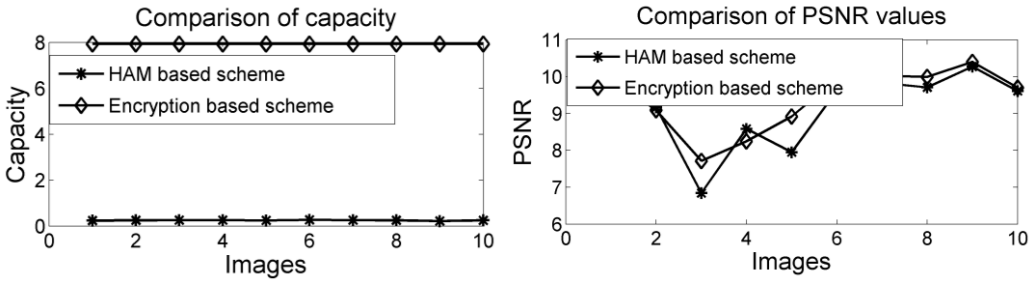
Table 9.2: The embedding times of the LBP and the LTP schemes

Schemes	Image									
	1	2	3	4	5	6	7	8	9	10
LBP	1.89	1.85	1.85	1.86	1.86	1.86	1.86	1.89	1.86	1.87
LTP with t=5	1.89	1.9	1.88	1.91	1.92	1.88	1.89	1.89	1.9	1.91
LTP with t=10	1.89	1.88	1.91	1.92	1.94	1.98	1.99	1.95	1.93	1.91

If the values in Table 9.2 is concatenated with the values of Table 9.1, the execution times of the schemes, which are discussed in this subsection, will be found. The values show that the LBP is the most optimal one regarding the execution time and the second optimal one is the LTP based method. The reason is that the schemes other than the LBP and the LTP first generate a prediction value by applying prediction rules on a set of neighbor pixels and then generate prediction errors before the start of the data implantation process. The prediction method has to compute several affairs for each pixel which takes much time, e.g., the location of the predicting pixel, the number of associated neighbors and the required operations with the associated pixels like the mean of the block pixels, the gradient of a pixel. On the other hand, the LBP and the LTP do not apply any prediction process, rather, these two schemes generate codes comparing the pixel values with a reference value in a single step and then generate encoded prediction values in another step. Thus, the LBP and the LTP methods spend less amount of time while completing the embedding task.

9.2.4 The Schemes which Destroy the Image Quality Intentionally

Intentionally image distortion based reversible embedment schemes are useful when the cover image itself is a secret. These schemes fully destroy the cover information in the stego image either during the bit implantation, e.g., the scheme in Chapter 7, or before the start of the bit implantation process, e.g., the scheme in Chapter 8. The scheme in the Chapter 7 destroys the image quality by the applied embedding rules, whereas the scheme in the Chapter 8 razes the image quality by using an encryption method to the image contents. In the following, the comparisons of their achieved embedding capacities and PSNR values are demonstrated.



(a) (b)

Figure 9.7: Comparisons of (a) embedding capacity and (b) PSNR value of the image distortion based schemes.

The embedding capacities and PSNR values for ten images are presented in the Figure 9.7(a) and Figure 9.7(b), respectively, for the HAM and the encryption based schemes. The embedding capacity in the encryption based scheme is 8bpp. The proposed encryption based scheme provides very unlike and higher embedding capacity. The PSNR values are very small and very close to each other. Therefore, in hiding massive data, the proposed encryption based scheme will be a better choice.

Table 9.3: The embedding times of the HAM and the encryption based schemes

Schemes	Image									
	1	2	3	4	5	6	7	8	9	10
HAM	7.3	7.32	7.54	7.43	7.33	6.9	7.1	7.4	6.9	7.1
Encryption	38.72	40.16	40.11	40.23	40.29	40.54	40.33	40.23	40.03	38.25

Again, if the size of the to-be-implanted data is too big and the data have to be implanted into more than one image, the encryption based method will not provide better performance

because the later scheme takes much time in its bitwise encryption operation, as shown in Table 9.3.

9.3 Summary and Comments

The thesis proposes several embedding methods each of which demonstrates outstanding performance regarding their competing schemes in their embedding area. These schemes are presented in Chapter 3 to Chapter 8. The performed comparisons of the proposed schemes, in this chapter, reveal that the LBP based scheme provides higher embedding payload and higher payload per PSNR value. Nevertheless, the scheme produces big assistant information which is proposed to be sent to the destination through another communication channel. The multi-predictor based scheme, on the other hand, do not send any assistant information through another channel; however, it presents large embedding payloads and significantly higher value of PSNR. When the size of the to-be-implanted data is very small compared to the image size, the LTP based scheme provides outstanding performance regarding the image quality. The control parameter t, defined in Chapter 6, is a very useful parameter to manage the image quality according to the demanded payloads of the application. Between the two image distortion based schemes, the encryption based one shows incomparable better performance. However, the investigation states that the encryption based scheme takes much time and hence, the scheme will not be a useful one if it is implemented in an online based application that requires to implant the message bits into more than one image.

Chapter 10

Conclusions and Future Works

The use of image steganography is continuously increasing for the purpose of covertly communication of secret information. Between irreversible and reversible image steganographic methods, the reversible processes exhibit more implementation challenges and provide ever-increasing supports of being used for practical applications. The reversible data hiding schemes devote their concentration mainly on improving either the embedding capacity [36], the stego image quality [53] or both [51]. Several latest schemes [52, 71, 110] intentionally destroy their stego image to provide better security to the cover contents when the cover image itself is secret. During the achievement of these major goals, the proposed schemes take care to minimize the time complexity and to improve the security of the implanted data. The embedding capacity depends on several issues. The notable matters are:

i) the correlations among the pixels in each locality.

ii) uses of embedding space, e.g., spatial domain, transformed coefficients, neighbour pixel differences and prediction errors.

iii) applied method to generate the embedding space, e.g., types of predictor,

iv) applied embedding rules.

v) tolerance of the system to the level of image distortions observed during the data implantation by the embedding rules.

The level of image distortion is governed by the demand of embedding capacity and the highest amount of pixels' displacement that is allowed by the embedding rules. The time complexity is a concern issue when data communication between two parties should be done within a time constraint, e.g., in live communications and steganographic based authentication processes. Though the thesis does not directly contribute to the issue of reducing the execution time of the proposed steganographic processes, the schemes implemented in the Chapter 5 and 6 dominate their competing schemes by the time complexity. The security of the implanted data is mainly provided by encrypting image, assigning negotiated values to secret parameters,

e.g., starting point of the data embedment task in an image, length of the secret message and initialized values of the parameters at the pre-processing stage. Though the thesis improves the security of the implanted data as discussed in Chapters 3 to 8 at its pre-processing stage, it shows a novel mechanism of implanting security at the post-processing stage in Chapter 8 by applying encryption process after the completion of the embedment task.

Taking all of these issues into consideration, the thesis proposes numerous predictive reversible data hiding processes where one or several predictors are applied to predict pixels. The thesis either improves the prediction accuracy of an existing scheme to improve the frequencies of the embeddable errors (e.g., the Chapters 3 and 4), uses the prediction errors in more efficient way to enhance the embedding capacity and the stego image quality (e.g., the Chapter 5) rather than concentrating to improve the prediction accuracy, employs encoded error based scheme to embed into all the image pixels (e.g., the Chapter 6) or shows an way of applying predictor to the existing non-prediction based new data embedment policies (e.g., the Chapters 7-8) to enhance the embedding capacity.

The major contributions of the thesis are:

- the development of a multi-block centre reference predictor that improves the prediction accuracy with compared to its competing scheme for the purpose of enhancing the quantity of the embeddable errors (e.g., the Chapter 3).
- the formulation for the issue of block centre biasness of the block pixels to improve the prediction accuracy (e.g., the Chapter 3).
- the development of a new predictor with higher accuracy by employing multiple predictors owing to improve the frequency of two embeddable errors in the single layer data embedment process (e.g., the Chapter 4).
- the implementation of a multilayer multi-cycle scheme where the embedding capacity per structural dissimilarity index is improved notably by embedding into fewer sample errors for multiple times rather than more sample errors for single time (e.g., the Chapter 5).
- the development of a new encoded error based data embedment process where local binary pattern and local ternary pattern code are used both to improve the image quality and to achieve higher embedding capacity as well as to control the

quantity of generated embeddable errors depending on the demanded embedding capacity (e.g., the Chapter 6).

- the implementation of histogram association and mapping based data embedment scheme by the help of prediction error histogram rather than the presently operational pixel value histogram to improve the embedding capacity as well as to distort the image quality intentionally (e.g., the Chapter 7).
- the development of a method that is able to implant up to 8 bits per pixel in the intentional image distortion based embedding category.

The proposals vary for the applied predicting methodologies, uses of the embedding layer, applied repeated embedment in the prediction errors in the quality preservation based embedding area and uses of predictor in the quality distortion based area. The experiments are conducted on several image datasets to test the proposals. The category wise results are presented in this thesis as chapter by chapter. The results presented in the depicted figures and tables are promising. All the results in their own categories noticeably dominate the other competing schemes. Though all the processes demonstrate more or less improved results, some of the improvements are very attractive as shown in Chapters 4, 6, 7 and 8. The scheme of the Chapter 4 demonstrates the process of using multiple predictors to enhance the quantity of embeddable prediction errors. The scheme presented in Chapter 6 is unbeatable because it ensures both the image quality and higher embedding capacity through implanting bit into every pixel. Chapter 7 presents an embedment process that uses prediction error histogram rather than using pixel value histogram, while implanting histogram association and mapping policy in its data embedment task. Though, in this thesis, several methods are proposed and implemented to enhance the embedding capacity as well as the stego image quality, the author believes that there are many potential areas where the research findings presented could be extended:

- **Demolishing the necessity of side-information:** Most of the embedding schemes use side information in order to de-embed secret message from the stego image at the receiver end. The side-information is compressed first and then, either implanted into the cover image in a specific part through LSB substitutions or sent to the destination through another communication channel. In the case of implanting the side information in the cover image, the pure embedding capacity is decreased. If the side information

used in different methods can be demolished, the pure embedding capacity can be increased significantly.

- **Applying LTP codes in multi-layer embedding process:** The schemes proposed, in this thesis, apply LTP codes in single-layer data embedment process. These LTP codes can also be used in the multi-layer embedding process to improve the image quality and to meet the requirement of embedding capacity dynamically.
- **Reducing image distortions:** The data embedment processes stated in Chapters 3, 4 and 6 try to implant a single bit in each pixel. It is possible to attain an image quality of 57 dBm or higher if a group of bits can be implanted in a block e.g., 9 bits in a block of 9 pixels, at a time by modifying just a single pixel in the block. Modelling of such a scheme can be explored in future.
- **Specific application based development:** E-medicine, m-medicine, forensic and law-enforcing agencies require different stego image management process. For these purposes, the embedding process varies. Design and development of application specific embedding scheme will be a major contribution in future research.

The presented contributions of this thesis will make significant effects on the field of reversible data hiding arena. These contributions have boosted up the embedding capacity notably, which is a crying demand in the current high volume data communication world. The proposed image quality preservation based schemes present higher image quality than their competing schemes. The LBP and the LTP based embedment processes are two unique policies for enhancing both the embedding capacity and the image quality. These two schemes will attract the attention of the application developers and the researchers in this area.

Bibliography

[1] Andriotis P., Oikonomou G. and Tryfonas T., "JPEG steganography detection with Benford's law", *Digital Investigation,* 9(3-4): 246-257, March 2013.

[2] Arshadi L. and Jahangir A. H., "Benford's law behavior of Internet traffic", *Journal of Network and Computer Applications,* 40: 194-205, April 2014.

[3] Böhme R. and Kirchner M., "Counter-forensics: attacking image forensics", *Digital Image Forensics*, Springer New York, ISBN 978-1-4614-0756-0, pp. 327-366, 2013.

[4] BOSS: Bank of standardized stimuli, https://drive.google.com/drive/folders/0B3m1Sf 0USgt8b1VET3NLcTVIc0U, last visited: 16 January 2017.

[5] Brindha S., and Vennila I., "Hiding fingerprint in face using scattered LSB embedding steganographic technique for smart card based authentication system", *International Journal of Computer Applications,* 26(10): 51-55, July 2011.

[6] CalTech101: Images of category of 101 of computer vision lab of the California Institute of Technology, USA, https://www.vision.caltech.edu/Image_Datasets / Caltech101 / 101_ObjectCategories.tar.gz, last visited: 16 January 2017.

[7] Chang I. C., Hu Y. C., Chen W. L. and Lo C. C., "High capacity reversible data hiding scheme based on residual histogram shifting for block truncation coding", *Signal Processing*, 108(C): 376-388, March 2015.

[8] Chao R. M., Wu H. C., Lee C. C. and Chu Y. P., "A novel image data hiding scheme with diamond encoding", *EURASIP Journal on Information Security,* 1(2009): 658047, May 2009.

[9] Chen C. C. and Tsai Y. H., "Adaptive reversible image watermarking scheme", *Journal of Systems and Software,* 84(3): 428-434, March 2011.

[10] Chen P. Y., Cheng S. M. and Chen K. C., "Information fusion to defend intentional attack in Internet of things", *IEEE Journal of Internet of Things*, 1(4): 337-348, August

2014.

[11] Chen X, Sun X, Sun H., Zhou Z. and Zhang J., "Reversible watermarking method based on asymmetric-histogram shifting of prediction errors", *Journal of Systems and Software,* 86(10): 2620-2626, October 2013.

[12] Cheng C. J., Hwang W. J., Zeng H. Y. and Lin Y. C., "A fragile watermarking algorithm for hologram authentication", *Journal of Display Technology*, 10(4): 263-271, April 2014.

[13] Chung K. L., Huang Y. H., Yan W. M. and Teng W. C., "Distortion reduction for histogram modification-based reversible data hiding", *Applied Mathematics and Computation,* 218(9): 5819-5827, January 2012.

[14] CRISP-Satellite: Centre for remote imaging, sensing and processing, National University of Singapore, https://crisp.nus.edu.sg/coverage/S_NPPindex.php, last visited: 16 January 2017.

[15] Ex-NSA contractor stole 50 TB of classified data; including top-secret hacking tools, *The Hacker News*, http://thehackernews.com/2016/10/nsa-contractor-hacking.html, last visited: 20 January 2017.

[16] Fang Y., Zeng K., Wang Z., Lin W., Fang Z. and Lin C. W., "Objective quality assessment for image retargeting based on structural similarity", *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 4(1): 95-105, March 2014.

[17] Feinstein L., Schnackenberg D., Balupari R. and Kindred D, "Statistical approaches to DDoS attack detection and response", in *Proceedings of DARPA Information Survivability Conference and Exposition*, 1: 303-314, 22 April 2003.

[18] Forouzan B. A., *Cryptography and network security*, Tata McGraw-Hill Publishing Company Limited, ISBN 13: 978-007-066046-5, pp. 2-3, 2007.

[19] Fridrich J. and *Kodovsky J., "Rich models for st*eganalysis of digital images", *IEEE Transactions on Information Forensics and Security,* 7(3): 868-882, June 2012.

[20] Fridrich J., Goljan M. and Soukal D., "Higher-order statistical steganalysis of palette images", in *Proceedings of SPIE conference on Security and Watermarking of Multimedia Contents V*, 5020: 178-190, June 2003.

[21] Fridrich J.., Pevný T. and Kodovský J., "Statistically undetectable jpeg steganography:

dead ends challenges, and opportunities", in *Proceedings of the 9th workshop on Multimedia and security*, 3-14, Dallas, Texas, USA, 20-21 September 2007.

[22] Fu D. S., Jing Z. J., Zhao S. G. and Fan J., "Reversible data hiding based on prediction-error histogram shifting and EMD mechanism", *AEU-International Journal of Electronics and Communications*, 68(10): 933-43, October 2014.

[23] Fu D., Shi Y. Q. and Su W., "A generalized Benford's law for JPEG coefficients and its applications in image forensics", in *Proceedings of SPIE conference on Security and Watermarking of Multimedia Contents IX,* 6505, 27 February 2007.

[24] Fu Z.., Sun X. and Xi J., "Digital forensics of Microsoft Office 2007-2013 documents to prevent covert communication", *Journal of Communications and Networks*, 17(5): 525-533, October 2015.

[25] González-Lee M., Nakano-Miyatake M., Pérez-Meana H. and Sanchez-Perez G., "Script format document authentication scheme based on watermarking techniques", *Journal of Applied Research and Technology*, 13(3): 435-442, June 2015.

[26] Govind P. S. and Wilscy M., "A new reversible data hiding scheme with improved capacity based on directional interpolation and difference expansion", *Procedia Computer Science*, 46: 491-498, December 2015.

[27] Gudla S., Reyya S., Kotyada A. and Sangam A., "Key based least significant bit (LSB) insertion for audio and video steganography", *IEEE International Journal of Computer Science Engineering Research and Development* (IJCSERD), 3(1): 60-69, January-March 2013.

[28] Gujjunoori S. and Amberker B. B., "DCT based reversible data embedding for MPEG-4 video using HVS characteristics", *Journal of Information Security and Applications,* 18(4): 157-166, January 2013.

[29] Habiba S., Kamal A. H. M. and Islam M. M., "Enhancing the robustness of visual degradation based HAM reversible data hiding", *Journal of Computer Science*, 12(2): 88-97, March 2016.

[30] Hackers' $81 Million Sneak Attack on World Banking, https://www.nytimes.com /2016/05/01/business/dealbook/hackers-81-million-sneak-attack-on-world-banking.html ?_r=0, last visited: 20 January 2017.

[31] Hong W., "Adaptive reversible data hiding method based on error energy control and

histogram shifting", *Optics Communications,* 285(2): 101-108, January 2012.

[32] Hong W. and Chen T. S., "A local variance-controlled reversible data hiding method using prediction and histogram-shifting", *Journal of Systems and Software,* 83(12): 2653-2663, December 2010.

[33] Hong W. and Chen T. S., "A novel data embedding method using adaptive pixel pair matching", *IEEE Transactions on Information Forensics and Security,* 7(1): 176-184, February 2012.

[34] Hong W., Chen T. S. and Luo C. W., "Data embedding using pixel value differencing and diamond encoding with multiple-base notational system", *Journal of Systems and Software,* 85(5): 1166-1175, May 2012.

[35] Huang F., Luo W. and Huang J., "Steganalysis of JPEG steganography with complementary embedding strategy", *IET Information Security*, 5(1): 10-18, March 2011.

[36] Islam S.and Gupta P., "Effect of morphing on embedding capacity and embedding efficiency", *Neurocomputing*, 137: 136-141, August 2014.

[37] Jana B., Giri D. and Mondal S. K., "An Efficient Data Hiding Scheme using Hamming Error Correcting Code", Proceedings of the Sixth ACM International Conference on Computer and Communication Technology, pp. 360-365, 25 September 2015.

[38] Juels A., "RFID security and privacy: a research survey", *IEEE Journal on Selected Areas in Communications*, 24(2): 381-94, February 2006.

[39] Kamal A. H. M. and Islam M. M., "Boosting up the data hiding rate multi cycle embedment process", *Journal of Visual Communication and Image Representation*, 40: 574-588, July 2016.

[40] Kamal A. H. M. and Islam M. M., "Capacity improvement of reversible data hiding scheme through better prediction and double cycle embedding process", in *Proceedings of IEEE International Conference on Advance Networks and Telecommunication Systems* (ANTS), Kolkata, India, 16-18 December 2015.

[41] Kamal A. H. M. and Islam M. M., "Enhancing embedding capacity and stego image quality by employing multi predictors", *Journal of Information Security and Applications*, 32: 59-74, February 2017.

[42] Kamal A. H. M. and Islam M. M., “Enhancing the embedding payload by handling the affair of association and mapping of block pixels through prediction errors histogram”, in *Proceedings of International Conference on Networks Systems and Security* (NSysS), BUET, Dhaka, 5-8 January, 2016.

[43] Kamal A. H. M. and Islam M. M., "Enhancing the performance of the data embedment process through encoding errors", *Journal of Electronics*, 5(4): 79-95, November 2016.

[44] Kamal A. H. M. and Islam M. M., "Facilitating and securing offline e-medicine service through image steganography", *Healthcare Technology Letters,* 1(2): 74-79, June 2014.

[45] Kamstra L. and Heijmans H. J., "Reversible data embedding into images using wavelet techniques and sorting",*IEEE Transactions on Image Processing,* 14(12): 2082-2090, December 2005.

[46] Khan S., Gani A., Wahab A. W. A, Shiraz M. and Ahmad I, "Network forensics: review, taxonomy, and open challenges", *Journal of Network and Computer Applications*, 66: 214-35, May 2016.

[47] Koblitz A. H., Koblitz N and Menezes A., "Elliptic curve cryptography: the serpentine course of a paradigm shift", *Journal of Number Theory*, 131(5): 781-814, May 2011.

[48] Lee C. F. and Chen H. L., "A novel data hiding scheme based on modulus function", *Journal of Systems and Software,* 83(5): 832-843, May 2010.

[49] Lee J.D., Chiou Y. H. and Guo J. M., "Information hiding based on block match coding for vector quantization-compressed images", *IEEE Systems Journal,* 8(3): 737-748, September 2014.

[50] Lei P. A. N. G., SUN M. H., LUO S. S., Bai W. A. N. G. and Yang X. I. N., "Full privacy preserving electronic voting scheme", *The Journal of China Universities of Posts and Telecommunications*, 19(4): 86-93, August 2012.

[51] Leung H. Y., Cheng L. M., Liu F. and Fu Q. K., "Adaptive reversible data hiding based on block median preservation and modification of prediction errors", *Journal of Systems and Software,* 86(8): 2204-2219, August 2013.

[52] Liao X. and Shu C., "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, April 2015.

[53] Liao X., Wen Q. Y. and Zhang J, "A steganographic method for digital images with four-pixel differencing and modified LSB substitution", *Journal of Visual Communication and Image Representation*, 22(1): 1-8, January 2011.

[54] Lin C. C., Tai W. L. and Chang C. C., "Multilevel reversible data hiding based on histogram modification of difference images", *Pattern Recognition*, 41(12): 3582–3591, December 2008.

[55] Lin C. C., "An information hiding scheme with minimal image distortion", *Computer Standards and Interfaces,* 33(5): 477-484, September 2011.

[56] Liu J., Tang G. and Sun Y., "A secure steganography for privacy protection in healthcare system", *Journal of Medical Systems*, 37(2): 1-10, April 2013.

[57] Liu M., Seah H. S., Zhu C., Lin W. and Tian F., "Reducing location map in prediction-based difference expansion for reversible image data embedding", *Journal of Signal Processing*, 92(3): 819-28, March 2012.

[58] Liu Y., Li Z., Ma X. and Liu J., "A robust data hiding algorithm for H. 264/AVC video streams", *Journal of Systems and Software*, 86(8): 2174-2183, August 2013.

[59] Lu Y. Y. and Huang H. C., "Adaptive reversible data hiding with pyramidal structure", *Vietnam Journal of Computer Science,* 1(3): 1-13, August 2014.

[60] Lu Z. M., Wang J. X. and Liu B. B., "An improved lossless data hiding scheme based on image VQ-index residual value coding", *Journal of Systems and Software,* 82(6): 1016-1024, June 2009.

[61] Luo H., Yu F. X., Chen H., Huang Z. L., Li H. and Wang P. H., "Reversible data hiding based on block median preservation", *Journal of Information Sciences*, 181(2): 308–328, January 2011.

[62] Luo L., Chen Z., Chen M., Zeng X. and Xiong Z., "Reversible image watermarking using interpolation technique", *IEEE Transactions on Information Forensics and Security*, 5(1): 187-93, March 2010.

[63] Ma X., Pan Z., Hu S. and Wang L., "High-fidelity reversible data hiding scheme based on multi-predictor sorting and selecting mechanism", *Journal of Visual Communication and Image Representation,* 28: 71-82, April 2015.

[64] Maheswari S. and DJ H., "Frequency domain QR code based image steganography using

Fresnelet transform", *AEU-International Journal of Electronics and Communications*, 69(2): 539-44, February 2015.

[65] Malik A., Sikka G. and Verma H. K., "A high capacity text steganography scheme based on LZW compression and color coding", *International Journal on Engineering Science and Technology*, in press, http://dx.doi.org/10.1016/j.jestch. 2016.06.005, August 2016.

[66] Mobasseri B. G. and Lynch R. S., "Information embedding in sonar by modifications of time-frequency properties", *IEEE Journal of Oceanic Engineering*, 41(1): 139-154, January 2016.

[67] Murala S., Maheshwari R. P. and Balasubramanian R., "Local tetra patterns: a new feature descriptor for content-based image retrieval", *IEEE Transactions on Image Processing*, 21(5): 2874-2886, May 2012.

[68] Murdoch SJ and Lewis S, "Embedding covert channels into TCP/IP", *in International Workshop on Information Hiding*, 3727: 247-261, June 2005.

[69] Natural: *National History Museum Public Library*, UK, http://piclib.nhm.ac.uk/, last visited: 16 January 2017.

[70] Nguyen Thai-Son, Chang C. C. and Ngoc-Tu Huynh, "A novel reversible data hiding scheme based on difference-histogram modification and optimal EMD algorithm", *Journal of Visual Communication and Image Representation*, 33: 389-397, November 2015.

[71] Ong S. Y., Wong K. and Tanaka K., "A Scalable Reversible Data Embedding Method with progressive quality degradation functionality", *Journal of Signal Processing: Image Communication,* 29(1): 135-149, January 2014.

[72] Ou B., Li X., Zhao Y. and Ni R., "Reversible data hiding based on PDE predictor", *Journal of Systems and Software,* 86(10): 2700-2709, October 2013.

[73] Pevny T., Bas P. and Fridrich J., "Steganalysis by subtractive pixel adjacency matrix", *IEEE Transactions on Information Forensics and Security,* 5(2): 215-224, June 2010.

[74] Por L. Y., Wong K. and Chee K. O., "UniSpaCh: A text-based data hiding method using Unicode space characters", *Journal of Systems and Software*, 85(5): 1075-1082, May 2012.

[75] Provos N. and Honeyman P., "Hide and seek: An introduction to steganography", *IEEE*

Security and Privacy, 1(3): 32-44, May-June 2003.

[76] Qin C., Chang C. C. and Hsu T. J., "Reversible data hiding scheme based on exploiting modification direction with two steganographic images", *Journal of Multimedia Tools and Applications*, 74(15): 5861-72, August 2015.

[77] Rabizadeh M., Amirmazlaghani M. and Ahmadian-Attari M., "A new detector for contourlet domain multiplicative image watermarking using Bessel K form distribution", *Journal of Visual Communication and Image Representation*, 40: 324-34, October 2016.

[78] Roy S. and Venkateswaran P., "Online payment system using steganography and visual cryptography", in *Proceedings of IEEE Conference on Electrical, Electronics and Computer Science* (SCEECS), 1-5, 1 March 2014.

[79] Rubio Ó. J., Alesanco A. and García J., "Secure information embedding into 1D biomedical signals based on SPIHT", *Journal of Biomedical Informatics*, 46(4): 653-664, August 2013.

[80] Rura L., Issac B. and Haldar M. K., "Secure electronic voting system based on image steganography", *IEEE Conference on Open Systems* (ICOS), 80-85, 25 September 2011.

[81] Sachnev V., Kim H. J., Nam J., Suresh S. and Shi Y. Q., "Reversible watermarking algorithm using sorting and prediction", *IEEE Transactions on Circuits and Systems for Video Technology*, 19(7): 989-99, July 2009.

[82] Shaukat A., Chaurasia M. and Sanyal G., "A novel image steganographic technique using fast fourier transform", in *Proceedings of IEEE International Conference on Recent Trends in Information Technology* (ICRTIT), 1-6, 8 April 2016.

[83] Tai W. L., Yeh C. M. and Chang C. C., "Reversible data hiding based on histogram modification of pixel differences", *IEEE Transactions on Circuits and Systems for Video Technology,* 19(6): 906-910, June 2009.

[84] Tang M., Hu J. and Song W., "A high capacity image steganography using multi-layer embedding", *Optik-International Journal for Light and Electron Optics*, 125(15): 3972-3976, August 2014.

[85] Nguyen T. S., Chang C. C. and Xiao-Qian Yang, "A reversible image authentication scheme based on fragile watermarking in discrete wavelet transform domain.", AEU-International Journal of Electronics and Communications, 70(8): 1055-1061, 31 August 2016.

[86] Tian H., Zhou K., Jiang H., Liu J., Huang Y. and Feng D., "An M-sequence based steganography model for voice over IP", in *Proceedings of IEEE International Conference on Communications* (ICC'09), 1-5, 14 June 2009.

[87] Trump condemns CIA Russia hacking report, http://www.bbc.com/news/world-us-canada-38292392, last visited: 20 January 2017.

[88] Tsai P., Hu Y. C. and Yeh H. L., "Reversible image hiding scheme using predictive coding and histogram shifting", *Journal of Signal Processing,* 89(6): 1129-1143, June 2009.

[89] Ulutas M., Ulutas G. and Nabiyev V. V., "Medical image security and EPR hiding using Shamir's secret sharing scheme", *Journal of Systems and Software,* 84(3): 341-353, March 2011.

[90] USC-SIPI Standard: *Signal and Image Processing Institute, University of Southern California*, USA, http://sipi.usc.edu/database/database.php?volume=misc, last visited: 16 January 2017.

[91] USC-SIPI Texture: *Signal and Image Processing Institute, University of Southern California*, USA, http://sipi.usc.edu/database/database.php?volume=textures, last visited: 16 January 2017.

[92] Viswanathan P. and PV K., "A joint FED watermarking system using spatial fusion for verifying the security issues of teleradiology", *IEEE Journal of Biomedical and Health Informatics*, 18(3): 753-64, May 2014.

[93] Zhou W., Bovik A. C., "A universal image quality index", *IEEE Signal Processing Letters*, 9(3): 81-84, March 2002.

[94] Wang J., Ni J. and Hu Y., "An efficient reversible data hiding scheme using prediction and optimal side information selection", *Journal of Visual Communication and Image Representation*, 25(6): 1425-1431, August 2014.

[95] Wang W. J., Huang C. T. and Wang S. J., "VQ applications in steganographic data hiding upon multimedia images", *IEEE Systems Journal,* 5(4): 528-537, December 2011.

[96] Wang X. Y., Wang C. P., Yang H. Y. and Niu P. P., "A robust blind color image watermarking in quaternion Fourier transform domain", *Journal of Systems and Software*, 86(2): 255-277, February 2013.

[97] Wang X. T., Chang C. C., Nguyen T. S. and Li, M. C., "Reversible data hiding for high quality image exploiting interpolation and direction order mechanism", *Digital Signal Processing*, 23(2): 569-577, March 2013.

[98] Wang Z. H., Lee C. F. and Chang C. Y., "Histogram-shifting-imitated reversible data hiding", *Journal of Systems and Software,* 86(2): 315-323, February 2013.

[99] Wang Z., Bovik A. C., Sheikh H. R. and Simoncelli E. P., "Image quality assessment: From error visibility to structural similarity", *IEEE Transactions on Image Processing*, 13(4): 600-612, April 2004.

[100] Weng C. Y., Zhang Y. H., Lin L. C. and Wang S. J., "Visible watermarking images in high quality of data hiding", *Springer Science and Business Media*, 66(2): 1033-1048, November 2013.

[101] White paper: *Cisco VNI Forecast and Methodology*, 2015-2020, http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html, last visited: 03 October 2016.

[102] World Internet Users and 2016 Population Stats, http://www.internetworldstats.com/stats.htm, last visited: 03 October 2016.

[103] Xiong J., Li F. and Liu J., "Fusion of different height pyroelectric infrared sensors for person identification", *IEEE Sensors Journal*, 16(2): 436-46, January 2016.

[104] Xu Bo, Jia-zhen Wang and De-yun Peng, "Practical protocol steganography: hiding data in IP header", in *Proceedings of First Asia International Conference on Modelling and Simulation* (AMS), 584-588, 27-30 March 2007.

[105] Yan Z., Zhang P. and Vasilakos A. V., "A survey on trust management for Internet of things", *Journal of Network and Computer Applications*, 42: 120-34, June 2014.

[106] Yang C. H., Weng C. Y., Wang S. J. and Sun H. M., "Varied PVD+ LSB evading detection programs to spatial domain in data embedding systems", *Journal of Systems and Software*, 83(10): 1635-1643, October 2010.

[107] Yang W. J., Chung K. L., Liao H. Y. M. and Yu W. K., "Efficient reversible data hiding algorithm based on gradient-based edge direction prediction", *Journal of Systems and Software,* 86(2): 567-580, February 2013.

[108] Yee L., Wong K. and Chee K. O., "UniSpaCh: A text-based data hiding method using

unicode space characters", *Journal of Systems and Software*, 85(5): 1075-1082, May 2012.

[109] Zhang J., Ho A. T., Qiu G. and Marziliano P., "Robust video watermarking of H. 264/AVC", *IEEE Transactions on Circuits and Systems II: Express Briefs*, 54(2): 205-9, February 2007.

[110] Zhang X., Qian Z., Feng G. and Ren Y., "Efficient reversible data hiding in encrypted images", *Journal of Visual Communication and Image Representation,* 25(2): 322-328, February 2014.

[111] Zhao H., Wang H. and Khan M. K., "Difference histogram analysis of several reversible data hiding schemes", *IEEE Conference on Postgraduate Research in Asia Pacific in Microelectronics and Electronics*, 201-204, 19 January 2009.

[112] Zhao H., Wang H. and Khan M. K., “Statistical analysis of several reversible data hiding algorithms”, *Multimedia Tools and Applications*, 52(2): 277-290, April 2011.

[113] Zhenfei Z., Luo H., Lu Z. M. and Pan J. S., "Reversible data hiding based on multilevel histogram modification and sequential recovery", *AEU-International Journal of Electronics and Communications*, 65(10): 814– 826, October 2011.

[114] Zhibin P., Hu S., Ma X. and Wang L., "Reversible data hiding based on local histogram shifting with multilayer embedding", Journal of Visual Communication and Image Representation, 31: 64-74, 31 August 2015.

==========