STUDY ON REVERSIBLE DATA HIDING TECHNIQUES FOR IMAGES

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

In

Mathematics

by

Sanjay Kumar

(2K18/PHDAM/14)

Under the Supervision of

Prof. Anjana Gupta

(Delhi Technological University)

&

Dr. Gurjit Singh Walia

(SAG, DRDO, Delhi)

DEPARTMENT OF APPLIED MATHEMATICS
DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

BAWANA ROAD, NEW DELHI-110042, INDIA.

June, 2025

ACKNOWLEDGEMENTS

First and foremost, I thank the Almighty for giving me the strength and inspiration to carry out this research work. I owe a deep sense of gratitude to all of his comprehensive soul, whose divine light has enlightened my path throughout the journey of my research.

I would like to express my sincere and heartfelt thanks to my research supervisor, **Prof. Anjana Gupta**, for her valuable guidance, enthusiastic encouragement, and persistent support. I am truly grateful from the core of my heart for her meticulous approach, the wonderful assistance of her perspective, and fruitful discussions on the research topic. Her careful supervision and personal attention have given me a lot of confidence and enthusiasm during the different stages of my doctoral investigations.

I place on record my heartfelt gratitude and sincere thanks to **Dr. Gurjit Singh Walia**, who has been my supervisor, advisor, and mentor. He is the one whose expertise in the field is widely acclaimed. I thank him for his valuable advice, constant support, and revered guidance. Invariably, I fall short of words to express my sincere gratitude for his patience and motivation.

I express my sincere gratitude to **Prof. R. Srivastava**, Head of the Department of Applied Mathematics, for his endless support and cooperation.

I am deeply grateful to my current organization, Scientific Analysis Group, Delhi, for showing faith in me and extending cooperation during the process of carrying out my research work along with my professional responsibilities in the organization. I am thankful to the Director of SAG and all employees of SAG, Delhi, for their kind help and support throughout my research.

I dedicate this thesis to my grandfather, **Baba Parsade**, for his endless love, support, care, encouragement, and blessings throughout my academics. My father, **Sh. Jaswant Singh** not only raised and nurtured me but also taxed himself dearly over the years for my education and intellectual development. My mother, **Smt. Bhoti Devi**, my grandmother, late **Smt. Kishan Dae**, my brother, late **Sh. Sanjay** and my sister, **Sunaina**, have been the sources of motivation and strength during moments of despair and discouragement. I am also thankful to my brother **Sh. Ajay Kumar** for his immense cooperation.

Date: **(SANJAY KUMAR)**

Place: Delhi, India.

CANDIDATE'S DECLARATION

I, Sanjay Kumar (2K18/PHDAM/14), a Ph.D. student in the Department of Applied Mathematics, hereby declare that the thesis entitled "Study on Reversible Data Hiding Techniques for Images" which is submitted by me to the Department of Applied Mathematics, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of "Doctor of Philosophy in Mathematics", is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship, or other similar title or recognition.

Place: Delhi **(Sanjay Kumar)**

Date: 2K18/PHDAM/14

CERTIFICATE

Certified that **Mr. Sanjay Kumar** (2K18/PHDAM/14), has carried out his research work presented in this thesis entitled **"Study on Reversible Data Hiding Techniques for Images"** for the award of **Doctor of Philosophy in Mathematics** from Delhi Technological University, New Delhi, under our supervision. The thesis embodies the results of the original work and studies carried out by the student himself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or anybody else from this or any other University/Institution.

(Prof. Anjana Gupta)
Supervisor and Professor
Department of Applied Mathematics
Delhi Technological University
New Delhi, India

(Dr. Gurjit Singh Walia)
Supervisor and Scientist-F
Scientific Analysis Group
DRDO
New Delhi, India

(Prof. R. Srivastava)
Professor and Head
Department of Applied Mathematics
Delhi Technological University
New Delhi, India

Date:

This thesis is dedicated to my grandfather ***Baba Parsade.***

for his endless love, support and encouragement.

Abstract

With the advent of digitalization, a large amount of media data is being digitalized, and digital storage stores data in a discrete form, which constitutes a part of media that is represented by some discrete value, thus making them vulnerable to various attacks. Protection of data integrity is a major problem in the field of information security. Thus, cryptography and Information hiding mitigate these issues. In a cryptography-based system, the main idea is to change the data into an incomprehensible manner, but at least it reveals the message's presence. Steganography helps to overcome the shortcomings of cryptography by hiding information in any digital media so that it is impossible for an individual to notice, but the digital media becomes useless when the secret gets revealed due to the damage caused by the data. However, most of the data hiding techniques suffer from permanent distortion of cover which makes their usage questionable in certain fields like defence, military image, medical diagnosis, digital forensics, stereo image coding etc, which cannot tolerate any distortion, as the retrieval of the cover media is as important as the hidden data. This is when the reversible data hiding method comes into picture.

Reversible data hiding (RDH) is an innovative technique for data hiding that allows for the embedding of secret information into digital media such as images, videos, or audio files, with the crucial requirement that the original media can be perfectly restored after the hidden data is extracted. The key distinction of RDH lies in its ability to achieve lossless embedding, where the host medium can serve dual purposes, i.e., securely carry the hidden data and retain its original form upon data extraction.

In the plain domain, the data hiding process is carried out directly on the original, unencrypted host medium. In the encrypted domain, the host medium is encrypted before data hiding, ensuring that the original content remains confidential even during the embedding process.

However, most of the techniques suffer due to the concerns about low data capacity, data security, and mainly in handling the communication overhead. To overcome these issues, a multistage high capacity reversible data hiding technique without overhead has been de-

veloped where histogram peaks are exploited for embedding the secret data along with overhead bits both in the plain domain and encrypted domain, thus enhancing the embedding capacity, data security, lower bandwidth, and lesser time complexity.

To further enhance the embedding capacity, communication overhead, and data security, a more complex yet efficient technique has been developed where both left peak and right peak from the main histogram peak have been exploited for data embedding.

Most of the RDH techniques being interdependent, and the image recovery is possible only after the secret extraction processes. In an encrypted domain, access flexibility being a factor, the stakeholders allied will have different levels of access to the data. To ensure privacy and security to both data and cover image, a separable reversible data hiding method in an encrypted image employing bit pair difference at the pixel level has been developed. With an encrypted image containing data, the receiver can either extract secret information, decrypt the cover media, or do both, depending on the available keys. Thus, the spatial correlation in natural images won't get exploited, ensuring high data embedding capacity. In the developed method, encrypted data is embedded in the cover image, therefore enhancing the security of the secret data along with the cover image.

It is very critical when it comes to secure communication of sensitive data over the public network, especially in the private or public 5G/6Gcellular network with tremendous deployment of the Internet of Things. Nowadays, most of the traffic over the communication channel is encrypted, and applications ensure end-to-end encryption of user data. Accordingly, solutions for data hiding over encrypted channels are invented that exploit redundant information or correlation among blocks of encrypted data for single user secure hiding of data. Thus, most of these communication networks and resources are underutilized as they offer high capacity transmission at multi-gigabit-per-second data rates. Therefore, a method has been developed where the user's secret data is hidden over encrypted traffic at the byte level without any communication overhead. The method is independent of the type of cover and does not exploit the correlation among the traffic bytes. Here, embedding is possible, multiple times in a single traffic byte, thus enhancing the privacy and security of the data. The developed method allows multiple users to communicate over a common channel without affecting individual users; thus, effective utilization of the channel bandwidth and the resources can be attained.

PSNR, MSE, POSP, NCC, and SSIM metrics are used to assess the imperceptibility of the image. The experimental validations over different types of cover have confirmed that the proposed techniques outperform the existing state-of-the-art techniques.

List of Publications

1. **Sanjay Kumar**, Anjana Gupta, G.S Walia; *Reversible data hiding: A contemporary survey of state-of-the-art, opportunities and challenges*, Applied Intelligence, Springer, Vol. 52, pp. 7373-7406, (2022), https://doi.org/10.1080/13682199.2024.2430874.

2. **Sanjay Kumar**, G.S. Walia, Anjana Gupta; *Multistage High-Capacity Reversible Data Hiding Technique Without Overhead Communication*, Defence Science Journal, Vol. 73, pp. 688-698, (2023), https://doi.org/10.14429/dsj.73.18991.

3. **Sanjay Kumar**, G.S. Walia, Anjana Gupta; *An Efficient Technique for Reversible Data Hiding using Bidirectional Histogram Shifting and Multistage Embedding*, Engineering Applications of Artificial Intelligence, Vol. 156, Art. 110986, (2025), https:doi.org/10.1016/j.engappai.2025.110986.

4. **Sanjay Kumar**, Anjana Gupta, G.S. Walia; *Dual Image Reversible Data Hiding: A Survey*, In 2023 IEEE Third International Conference on Secure Cyber Computing and Communication (ICSCCC), pp. 190-195. IEEE, (2023), 10.1109/ICSCCC58608.2023.10176708.

5. **Sanjay Kumar**, G.S. Walia, Anjana Gupta; *An Efficient Separable Reversible Data Hiding Method based on Bit-Pair Difference at pixel level*, The Imaging Science Journal, pp. 1-14, (2024), https://doi.org/10.1080/13682199.2024.2430874.

Patent Publication

- **Sanjay Kumar**, G.S. Walia, Anjana Gupta; *Method for Adaptive User Message Communication Over Encrypted Channel Traffic and System Thereof*, Patent file has been submitted.

Papers in International Conferences

- Presented a research paper entitled "**Dual Image Reversible Data Hiding: A Survey**" on 3^{rd} International Conference on Secure Cyber Computing and Communication (ICSCCC) held at NIT Jalandhar, India during September May 26-28, 2023.
- Presented a research paper entitled "**Some Investigation on Security Analysis of Reversible Data Hiding Techniques** " in 1^{st} *International Conference on "Algorithms, Computing and Systems",* held online by , National Institute for engineering Research, held on $10th$ September at Srinagar India, 2023.

Contents

List of Tables

List of Figures

List of Abbreviations

RDH	Reversible Data Hiding
RDHEI	Reversible Data Hiding in Encrypted Image
PSNR	Peak Signal to Noise Ratio
NCC	Normalized Cross Correlation
POSP	Proportion of shifted Pixels
MSE	Mean Square Error
dB	Decibel
BPP	Bit Per Pixel
EM	Embedding Capacity
LSB	Least Significant Bit
MSB	Most Significant Bit
SSIM	Structural Similarity Index
NMI	Neighbour Mean Interpolation
ED	Encrypted Domain
DE	Difference Expansion
PEE	Prediction Error-Expansion
NTRU	N^{th} degree Truncated polynomial Ring Units
VARE	Vacating Room After Encryption
VRBE	Vacating Room Before Encryption
MED	Median Edge Detector
GAP	Gradient Adjusted Predictor
G-LSB	Generalization Least Significant Bit
IDWT	Integer Discrete Wavelet Transform
RRAEUSE	Reserving Room After Encryption using Specific Encryption
RRAE	Reserving Room After Encryption
RRBE	Reserving Room Before Encryption
HS	Histogram Shifting

Chapter 1

Introduction

Data hiding is the process of embedding secret or additional information into a host medium, such as digital images, audio files, videos, or text, in a manner that ensures the hidden data remains imperceptible to unintended observers. This technique is widely used in digital watermarking, copyright protection, secure communication, and data authentication. The primary goal of data hiding is to provide a secure and efficient method of transmitting or storing information without arousing suspicion or degrading the host medium's quality. Reversible data hiding (RDH) is one such innovative technique for data hiding that allows for the embedding of secret information into digital media such as images, videos, or audio files, with the crucial requirement that the original media can be perfectly restored after the hidden data is extracted. The key distinction of RDH lies in its ability to achieve lossless embedding, where the host medium can serve dual purposes: securely carrying hidden data and retaining its original form upon data extraction. This characteristic sets RDH apart from conventional techniques and underscores its importance in domains that demand reversible modifications. The essence of RDH lies in its applications across various fields including military communications, medical imaging, and copyright protection, where maintaining the integrity of the original content is paramount. RDH enable users to securely share sensitive information while ensuring that the cover media remains unaltered after re-

covery. This method mitigates the risks associated with data loss and quality degradation, thereby strengthening the overall data security framework. In contemporary digital environments, where the need for data security and confidentiality is of utmost importance, RDH has become a focal point of research and development. The development of reversible data hiding can be traced back to the emergence of early steganography techniques in the 1990s. Initial approaches primarily aimed at embedding information while minimizing the perceptible changes in the host media. Over the years, researchers have introduced various algorithms and frameworks that enhance the effectiveness of RDH. These advancements have led to the classification of RDH methods into several categories, each with its unique techniques and applications.

Reversible Data Hiding (RDH) techniques offer significant benefits over traditional data hiding methods, primarily due to their unique capability to restore the original media after data extraction without any loss of quality. This section outlines the principal advantages of RDH in comparison with the conventional approaches.

- *Lossless Recovery of Original Media:* One of the most distinguishing features of reversible data hiding is the ability to perfectly recover the original cover media after the hidden data has been extracted. In traditional data hiding methods, alterations made to the host media often lead to permanent loss of quality or information. RDH ensures that the original state of the media can be restored, making it suitable for sensitive applications where integrity is paramount, such as medical imaging and military communications.

- *Enhanced Security Features:* Reversible data hiding inherently provides a higher level of security compared to traditional data hiding methods. RDH algorithms can incorporate encryption techniques, allowing the hidden data to be securely managed and

transferred. This dual-layer approach offers protection against unauthorized access or data tampering, making RDH an effective solution for confidential applications.

- *High Fidelity and Visual Quality Preservation:* Unlike traditional methods that may introduce noticeable artifacts or distortions in the altered media, RDH techniques generally maintain high visual quality. The aim is to embed data while ensuring minimal changes to pixel values, thereby reducing the likelihood of any perceptible degradation in the media quality. This characteristic is particularly crucial in fields where media quality is essential, such as in the restoration of images used in medical diagnostics.

- *Robustness Against Attacks:* Reversible data hiding methods are often designed to be robust against various forms of attacks, including compression, cropping, and noise interference. Traditional systems may allow for the degradation of the host signal under such transformations, but advanced RDH techniques implement strategies that preserve the embedded data even when the media undergoes transformations or compression. This robustness makes RDH suitable for use in diverse environments where media might be manipulated.

- *Flexibility in Data Embedding:* Reversible data hiding offers flexibility in terms of where and how data can be embedded within the host media. Various algorithms allow for different embedding strategies, which can be tailored to specific applications. This contrasts with traditional methods that may impose rigid structures on how and where data can be inserted and limiting their applicability.

- *Quantitative Performance Metrics:* RDH techniques often feature established quantitative performance metrics such as embedding capacity, imperceptibility and robustness. These metrics allow for systematic evaluations and comparisons with tradi-

tional methods, enabling researchers and practitioners to choose the most appropriate technique based on specific project requirements or constraints.

- *Application Versatility:* Due to their lossless nature, RDH methods are applicable across a wide range of sectors, including medical imaging, military communications, and digital rights management. Traditional methods often face limitations regarding the types of media they can effectively handle or the contexts in which they can be applied. RDH's versatility enables its adoption across various fields requiring secure and reliable data hiding solutions.

- *Facilitates Digital Watermarking:* Reversible data hiding techniques can effectively be used for digital watermarking, allowing copyright and ownership information to be embedded into digital content. The capacity to recover the original content makes RDH particularly attractive for watermarking applications, where the visibility of the watermark can be a concern. Traditional watermarking methods often compromise the visual integrity of the original content hindering their effectiveness.

- *Preservation of Sensitive Information:* In certain applications, preserving sensitive information is crucial. RDH methods can incorporate secret data directly into images or videos, enabling secure transactions while retaining the original media for future access. Traditional methods may not offer the same level of confidentiality, as they typically risk information loss during the process of data embedding. Reversible data hiding in the encrypted domain (RDH-ED) is the method used to safeguard the privacy of multimedia content in cloud services.

RDH techniques can be broadly categorized based on the domain in which they operate. These categories are briefly defined as follows.:

1.1 Plain Domain RDH

In plain domain, the data hiding process is carried out directly on the original, unencrypted host medium. Techniques in this category are widely used in scenarios where the host medium is available in its original form. The major methods include:

- *Histogram Shifting:* Histogram shifting is a widely adopted RDH technique that modifies the histogram of pixel intensity values in an image. The peak and zero points of the histogram are used to embed data by shifting pixel values in a reversible manner. Advantage is that its simple and efficient with minimal distortion. And challenge being limited embedding capacity and vulnerability to image quality variations.

- *Difference Expansion:* This method utilizes the differences between neighboring pixel values to embed data. These differences are expanded, creating room for data insertion while ensuring reversibility. Advantage is that it has got high embedding capacity. Challenges include, susceptible to distortions in high-contrast images.

- *Prediction Error Expansion:* It leverages the prediction errors derived from estimating pixel values based on their neighbors. By expanding these errors, data can be embedded in a reversible manner. Advantage is that it balances embedding capacity and visual quality. Requirement of careful modeling of prediction to minimize distortion is its challenge.

- *Compression-Based Techniques:* These techniques utilize the redundancy in the host medium by compressing parts of the data to make room for embedding. Advantages include effective in media with high redundancy. Computational intensity is what is to be worked on.

1.2 Encrypted Domain RDH

In the encrypted domain, the host medium is encrypted before data hiding, ensuring that the original content remains confidential even during the embedding process. This approach is especially relevant for secure communication and cloud storage applications. Techniques in this category include:

- *Vacating Room After Encryption (VRAE):* In VRAE, the host medium is encrypted first, and then the data embedding process creates space for data hiding. The reversibility is maintained through specific encryption and embedding strategies. Advantage is that it has high security as the original content remains encrypted. Limited embedding capacity and increased complexity is the main challenge faced.

- *Reversible Data Hiding in Encrypted Images (RDHEI):* RDHEI focuses on embedding data directly into encrypted images, ensuring that neither the embedded data nor the original image is exposed during transmission. Its advantage includes robust security for both data and image content. Challenge is that it requires careful coordination between encryption and data embedding processes.

These methods involve three parties namely content owner, the data hider and the receiver. The content owner is the original cover media owner who encrypts the image and sends it to the data hider. Data hider embeds data into the encrypted cover media but has no permission to access the original cover media. Receiver can decrypt the secret information without any in loss of secret information and cover media. Generally, RRBE creates some space into the cover media for data embedding, and then encrypts the cover media using encryption key. In RRAE approach, original cover media is encrypted by the content owner, and data hider alter the few bits of encrypted cover media for embedding

the information bit. These methods are mainly used for application such as cloud storage, secure remote sensing system, medical image management system and many more. However, embedding capacity of RRAE methods is limited due to the high entropy of encrypted image, and also secret information extraction and cover media recovery are not satisfactory. But, these approaches are used mainly due to simple and efficient operation at the end user.

In sum, a lot of RDH methods have been proposed in each category but there is still scope of improvement that can be considered for developing robust, high embedding capacity and improved visual quality of cover image. In addition, RDH methods also need to develop such techniques that need not share the communication separately overhead to extract the secret information and recover the cover image.

1.3 Performance Metrics

In this section, various performance metric to measure the reversible data hiding techniques are briefly discussed.

Embedding Capacity: is the maximum size of secure data that can be concealed in the cover image. It is measured in terms of bits per pixel (bpp) or total number of bits.

Mean-Squared Error: aims to compare between pixel values of original $I(i,j)$ and marked $I'(i,j)$ cover image of size $M \times N$ and it is determined by square of differences of the original and marked cover image. Mathematically, it is defined as

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} [I(i,j) - I'(i,j)]^2 \tag{1.3.1}$$

The value of MSE should be minimum and lie between $[0, \infty)$.

Peak Signal Noise Ratio: is a measure to calculate the quality between original and marked

cover image. It is measured in decibels (dB) and PSNR value is directly proportional to the quality of image and it is defined as follows:

$$PSNR = 10log_{10}\left(\frac{255^2}{MSE}\right) \tag{1.3.2}$$

Structural Similarity Index: is a quality assessment metric of marked cover image which determines the similarity between the original cover (I) and marked cover (I') image. The value of SSIM lies between [0,1] and it should be very close to one. This is determined as follows:

$$SSIM = \left(\frac{2\mu_I\mu'_I + c_1}{\mu_l^2\mu_I'^2 + c_1}\right)\left(\frac{2\sigma_{II'} + c_2}{\sigma_I^2 + \sigma_I'^2 + c_2}\right) \tag{1.3.3}$$

where μ_I, μ'_I and $\sigma_I^2, \sigma_I'^2$ are the averages and variances of original and marked cover images respectively. $\sigma_{II'}$ is the covariance of original and marked image, and C_1 and C_2 are the regularization constants.

Proportion of Shifted Pixels: is a ratio of shifted pixels (N_s) to sum of shifted and expanded $(N_s + N_e)$ pixels. This ratio should be a minimum because a high value of this ratio implies a high distortion in the marked cover image. Mathematically, it is defined as

$$POSP = \frac{N_s}{N_s + N_e} \tag{1.3.4}$$

where smaller value of POSP indicates that more data can be embedded with lesser distortion.

Normalized Cross Correlation: is used to measure the degree of similarity between the original (I) and marked(I') cover image. It is calculated as follows:

$$NCC = \frac{1}{M \times N}\sum_{i=1}^{M}\sum_{j=1}^{N}\frac{[I(i,j) - mean] \times [I'(i,j) - mean(I')]}{\sqrt{\sigma_I^2\sigma_I'^2}} \tag{1.3.5}$$

where $M \times N$ represents the size of the cover image and σ_I^2 & $\sigma_I'^2$ are the variances of the original and marked image, respectively. The value of NCC ranges from -1 to 1.

1.4 Thesis Overview

The thesis comprises five chapters, and a brief description of these chapters is given below:

Chapter 1:- This chapter covers the introduction to the topic of reversible data hiding. It will also contain the main idea for the development of the thesis and also an overview, and the objectives of the research work.

Chapter 2:- This chapter covers the state-of-the-art techniques developed in existing research work on “Reversible Data Hiding Techniques” and "Dual Image Reversible Data Hiding". It will also highlight the research gaps in the existing work that has stimulated the development of research objectives. In addition, benchmark datasets required for the performance validation of the RDH techniques are discussed.

Chapter 3:- This chapter highlights the details of the methodology adopted to accomplish the high-capacity multistage reversible data hiding techniques without overhead. It will also cover the reversible data hiding technique using bidirectional Histogram Shifting and Multistage Embedding. The observations and discussion of results from these two techniques will be presented as well.

Chapter 4:- This chapter highlights the details of the methodology adopted to accomplish the separable reversible data hiding method based on bit-pair difference at the pixel level. It will also cover the reversible data hiding technique over encrypted traffic at the byte level for secure data communication. A brief description highlighting the accuracy and the effectiveness of the proposed methodology will also be discussed.

Chapter 5:- This chapter contains a summary of all the ideas, observations, and contributions of the results obtained in each objective. Also, the future directions are sketched, and the social impact of reversible data hiding is also briefly discussed in this chapter..

1.5 Research Motivation

Reversible Data Hiding (RDH) has broadly gathered significant attention in recent years due to its unique ability to embed secret information into digital media such as images, videos, or audio by enabling faithful recovery of the original media after the embedded information is extracted. This capability of RDH makes it different from conventional data hiding techniques, making it especially appealing for applications where preserving the integrity of the original data is crucial. One of the most distinguished features of reversible data hiding is the ability to perfectly recover the original cover media after the hidden data has been extracted, whereas in traditional data hiding methods, alterations made to the host media often lead to permanent loss of quality and may also affect the information. RDH techniques can also be tailored to specific scenarios, ensuring optimal performance, making it prominent to use. One of the primary motivations for RDH includes maintaining integrity, alternatively preservation of sensitive information. In many practical applications, especially in fields like medical imaging and remote sensing, even minor alterations to the original data can have fatal consequences. Traditional methods may not offer the same level of confidentiality, as they typically risk information loss during the process of data embedding. RDH ensures that after the extraction of the embedded data, the original image can be perfectly restored, maintaining its accuracy and reliability. This makes RDH an invaluable tool in scenarios where data fidelity can't be compromised. Concerning sensitive data like metadata management, the RDH facilitates secure transactions. Another driving

force to opt for RDH is its enhanced security, where it inherently provides a higher level of security compared to traditional data hiding methods. RDH algorithms can incorporate encryption techniques, allowing the hidden data to be securely managed and transferred. This dual-layer approach offers protection against unauthorized access or data tampering, making RDH an effective solution for confidential applications, offering unparalleled security benefits. Traditional methods may introduce noticeable artifacts or distortions in the altered media, whereas RDH techniques generally maintain high visual quality. Hence, it may be referred preserving visual quality and high fidelity is another distinguishing factor that sets it apart from the traditional techniques. For example, in military and defence operations, embedding confidential information into satellite images or maps ensures that the data remains hidden while the integrity of the original content is preserved. The aim is to embed data while ensuring minimal changes to pixel values, thereby reducing the likelihood of any perceptible degradation regarding the quality of the media. Digital media is often subjected to processes like compression, cropping, or noise interference in cloud storage or social media. On the contrary, RDH methods are often designed to be robust against these forms of attacks. Advanced RDH techniques implement strategies that preserve the embedded data even when the media undergoes transformation or compression. Thus, the robustness makes RDH suitable for use in diverse environments where the conventional methods fail to preserve the integrity, due to which the media might be manipulated.

Another perspective that can be discussed is that RDH offers flexibility in terms of where and how data can be or has to be embedded within the host media. For that, the algorithms allow for different embedding strategies, which can be tailored to specific applications. For example, data can be embedded in selected regions of an image or distributed across multiple sections, depending on the application's needs. This contrasts with traditional

methods that may impose rigid structures that limit their practical utility and ensure compatibility with a wide range of use cases. RDH techniques allow copyright or ownership information to be embedded into digital content without degrading its visual quality, thus facilitating digital watermarking. Traditional watermarking methods often compromise the visual integrity of the original content, hindering their effectiveness where the visibility of the watermark can be a concern. This is when RDH plays its role. The motivations for RDH arise from the challenges and limitations of traditional data hiding methods, where lossless recovery, enhanced security, visual quality preservation, and adaptability are of major concern in today's growing digital world. As digital communication, storage, and processing continue to evolve, the role of RDH in ensuring data integrity and security will only become more prominent, along with its robust, flexible, and versatile nature.

1.6 Objective of Research Work

The main objective of this research work was the development of robust, reliable, and efficient reversible data hiding techniques for images. These specific objectives are summarized as follows:

Objective 1:

Review of state-of-the-art techniques and frameworks for reversible data hiding.

Objective 2:

- Design and Development of reversible data hiding techniques for plain domain.

- Experimental validation of the proposed framework on different datasets.

- Performance comparison of the proposed method with state-of-the-art techniques.

Objective 3:

- Design and Development of reversible data hiding techniques for the encrypted domain.
- To validate the proposed framework on different datasets.
- Performance comparison of proposed framework with state-of-the-art techniques.

1.7 Thesis Contribution

In this thesis, we show designed and developed reversible data hiding techniques in the plain and encrypted domains. These techniques solve the overhead issue, which requires sharing the overhead via a separate channel. Additionally, we also investigated the reversible data hiding technique, which embeds the secret data over encrypted traffic at the byte level. The main contribution of this thesis is:

- Comprehensive review of existing literature and classified the reversible data hiding methods mainly into a) Plain domain, b) Encrypted domain and also examine its pros and cons. A tabular comparison of various RDH methods has been provided, considering various design and analysis aspects. Moreover, we discuss important issues related to reversible data hiding and the use of benchmark datasets along with performance metrics for the evaluation of RDH methods.
- We first introduce a multistage high-capacity reversible data hiding technique without overhead in this thesis. The proposed reversible data hiding approach exploits histogram peaks for embedding the secret data along with overhead bits, both in plain

and encrypted domains. First, a marked image is obtained by embedding secret data in the plain domain, which is further processed using an affine cipher, maintaining correlation among the pixels. In the second stage, overhead bits are embedded in the encrypted marked image. High embedding capacity is achieved through exploiting the histogram peak for embedding multiple bits of secret data.

- Bidirectional embedding of secret data has been proposed wherein both left peak and right peak from the main histogram peak have been exploited for data embedding, thereby achieving high embedding capacity. Data embedding is performed at multistage both in the plain and encrypted domains, to achieve not only optimum quality but also high security. The secret data embedding capacity is also tunable, wherein the number of bits per pixel can be determined from the size of the cover and the amount of secret data. In addition, block-wise embedding of secret data further enhances the data embedding capacity.

- An efficient separable RDH method employing bit-pair differences has been proposed. Embedding at the bit pair level not only achieves high secret data embedding capacity but also enhances the security and privacy of the secret message and cover. The proposed method employs the average and floor function to generate the interpolated cover and hence, avoids the overflow/underflow problem. The proposed separable method supports blind data extraction from the marked cover, that is, secret messages and covers are recovered independently on the basis of available keys rather than sequence order. This method is computationally inexpensive as the embedding algorithm only considers simple rules based data hiding, and data retrieval is achieved through a simple XOR operation, which overcomes the communication overhead.

- At last, we introduce multi-user secret data embedding over the encrypted traffic without affecting the communication between end users. Ensuring faithful recovery of cover at each user is utmost required to develop the technology which can efficiently utilize the channel resources and also ensure privacy and security of multi-user data transmission over a common channel.

Chapter 2

Literature Review

2.1 Introduction

Reversible Data Hiding has been extensively explored due to its potential applications in the field of secure communication, such as copyright protection, source tracing, annotation of photographs, protection of data alteration, access control systems, content distribution, media database systems, and many more. The goal of this chapter is to review the state-of-the art Reversible Data Hiding and dual image RDH methods, classify these methods into different classes, and list out new trends in this field, to characterize the work and also to summarise plain domain and encrypted domain RDH methods, to analyse their computational complexity, merits, and demerits. A tabular comparison of various RDH methods has been provided, considering the different design and analysis aspects. Moreover, we discuss important issues related to reversible data hiding and the use of benchmarked datasets along with performance metrics for the evaluation of RDH methods. In addition, various authors have been broadly investigating dual image RDH methods to improve the data hiding capacity and security compared to ordinary RDH, as the former provides a good solution when compared to the latter. In this chapter, we have also classified dual

image RDH methods such as exploiting modification direction, pixel value differencing, central folding strategy, and turtle shell, along with explaining their merits and demerits. DI-RDH methods have been examined, considering data hiding capacity and PSNR. Performance of these methods is also considered in terms of data hiding capacity and quality of cover.

Over the past two decades, reversible data hiding has been extensively explored and reviewed. RDH techniques are mainly categorized into plain and encrypted domains. RDH methods in the plain domain have been proposed under lossless compression [1, 2, 3, 4], correlation expansion [5, 6, 7, 8], histogram shifting [9, 10, 11, 12, 13, 14], and interpolation [15, 16, 17, 18]. RDH methods in the encrypted domain have been proposed under, reserving room before encryption (RRBE) [19, 20, 21, 22] and reserving room after encryption(RRAE) [23, 24, 25]. The various RDH techniques under each category are briefly reviewed and are detailed as follows.

2.2 Proposed Classification of RDH Methods

Reversible data hiding approaches have been substantially investigated in the last two decades to achieve optimum recovery of the secret message and cover media. The cover media includes an image, a text file, audio, video, or multimedia. The proposed classification of the RDH techniques is depicted in Fig. 2.1.

Plain domain RDH framework is illustrated in Fig. 2.2 . In this, first the cover media is preprocessed to create space for data embedding in the cover media.

Further, secret information is embedded in the cover media using lossless embedding algorithm to obtain marked cover media. This process produces the marked cover media. Sender sends marked cover media to the recipient over the communication channel.

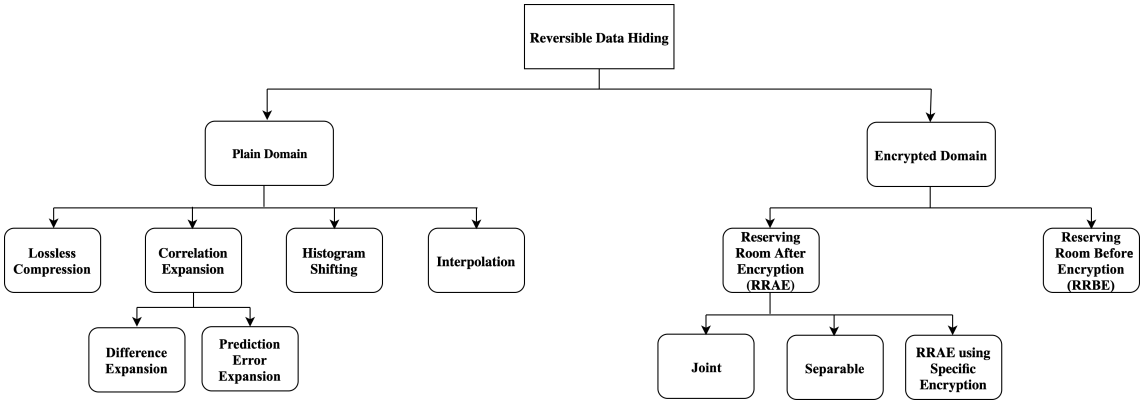

Fig. 2.1 Proposed Classification of Reversible Data Hiding Methods

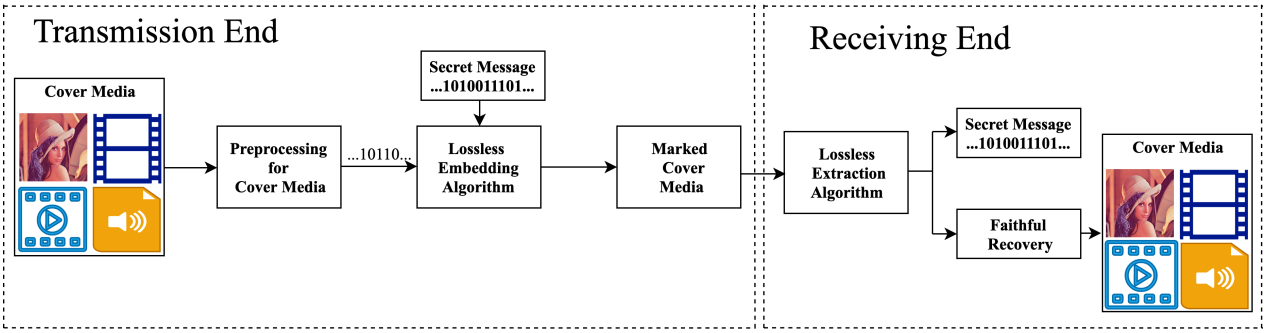

Fig. 2.2 General framework of Plain Domain RDH

The recipient retrieves secret information along with faithful recovery of the original cover media using lossless data extraction algorithm. Plain domain work can be categorized mainly into lossless compression [1, 2, 3, 4], correlation expansion [5, 6, 7, 8], histogram shifting [9, 10, 11, 12, 13, 14], and interpolation [15, 16, 17, 18]. In lossless compression, a part of the cover media is compressed for embedding data. These methods need more computing power, low data embedding capacity, and are hence used for the authentication of cover media. There is no loss of quality, but the size of the cover media is reduced due to compression. These methods are limitedly used due to low compression, time complexity, and a complex decoding procedure. In contrast, correlation expansion work can be classified as difference expansion [6, 7, 26] and prediction error expansion [8, 27, 28]. Difference expansion utilizes the correlation of two neighbouring pixels, while prediction error-expansion considers the local correlation of larger neighbouring pixels in the cover media to embed the secret data. In histogram shifting, secret information is em-

bedded at the peak of the histogram bin, thereby obtaining high embedding capacity and low computational complexity compared to other lossless compression approaches [9, 10]. In interpolation approaches, secret information is embedded into interpolated pixels without changing the original cover media pixels. The receiver can retrieve secret information from interpolated pixels, and the original cover media is recovered by scaling down the stego image. On the other hand, encrypted domain reversible data hiding methods emphasize the privacy and security of secret information along with the cover media. In order to decrypt the marked encrypted cover media, and to produce the retrieved cover media identical to the original, the receiver holds the choice to operate with different keys. The methodology of encrypted domain RDH is depicted in Fig. 2.3. Encrypted domain work involves three parties, namely the content owner, data hider, and receiver. Generally, the

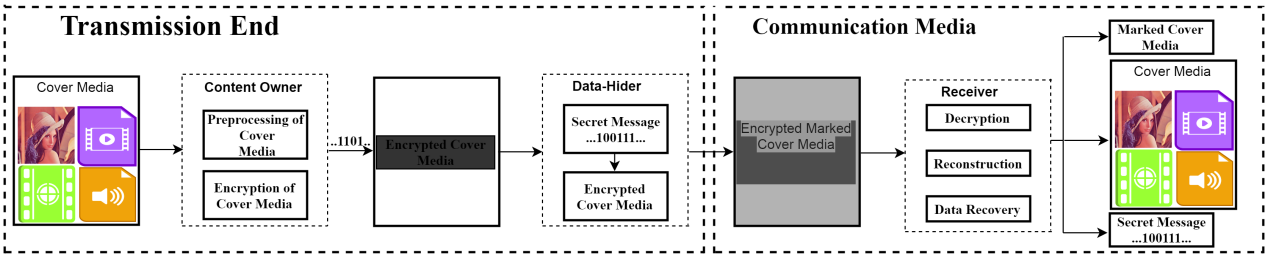

Fig. 2.3 General framework of Encrypted Domain RDH

cover media is either preprocessed or direct encrypted using the encryption key. Then, secret information is embedded into the encrypted cover media by the data hider and generates the marked encrypted cover media. This media is sent to the recipient over the communication channel. The receiver extracts the secret information along with the cover media to achieve privacy and security. Encrypted domain work is categorized mainly into Reserving Room After Encryption (RRAE), Reserving Room After Encryption using Specific Encryption(RRAEUSE), and Reserving Room Before Encryption (RRBE). In RRAE, correlation of the spatial domain is lost due to the encrypted cover media. Designing a general method is difficult for this method due to the loss of correlation in the spatial do-

main. Using the lossless extraction algorithm, the receiver decrypts the encrypted marked media using the decryption key, leading to the extraction of secret information along with the original cover media. Further, RRAE work is categorized as joint/non-separable and separable. In joint scheme, first encrypted marked media is decrypted, and after that secret information is extracted. In this, the receiver cannot extract secret information if the data hiding key is available without any knowledge of encryption key. Hence, this approach require data hiding key as well as encryption key for data extraction along with recovery of cover media. On the other hand, in separable, receiver can extract secret information, decrypt the cover media or both depending on his keys. In RRAUSE cover media is encrypted using specific encryption methods, which preserve some correlation among the pixels.

In RRBE, first free space is created in the cover media through preprocessing. Then, preprocessed media is encrypted using the encryption key and secret information is embedded into encrypted cover media. This encrypted marked media is sent to the receiver, where secret information is extracted along with original cover media through decryption key.

Plain domain and encrypted domain work has individual merits and demerits. But this work has been extensively explored to achieve high capacity without degradation in quality. Research work in these domains are further detailed in the following section.

2.3 Plain Domain Reversible Data Hiding

Plain domain approaches of RDH are aimed to achieve high payload capacity and tunable visual quality of cover media without any protection of cover media. We have categorized the plain domain work as a) lossless compression [1, 2, 3, 4, 29] b) correlation expan-

sion [5, 6, 7, 8] c) histogram shifting [9, 10, 11, 12, 13, 14], interpolation [15, 16, 17, 18]. Also, correlation work is further classified as difference expansion and prediction error expansion. Lossless compression methods are mainly used for authentication due to its limited embedding capacity. Correlation expansion approach utilizes the correlation of the spatial domain for hiding secret information in the cover media, while in histogram shifting, the peak of the histogram bin is used. The details of these methods, along with an analysis of the reported research work in these directions, are as follows.

2.3.1 Lossless Compression

Lossless compression based RDH approaches is mainly applied for authentication due to their low capacity. In this, free space is created and further used for data embedding in cover media. This approach provides limited data embedding capacity and is extensively explored mainly for cover media authentication. The performance depends on compression as well as a selected subset of cover media. An overview of the general framework for lossless compression is depicted in Fig. 2.4. In this framework, cover media is prepro-

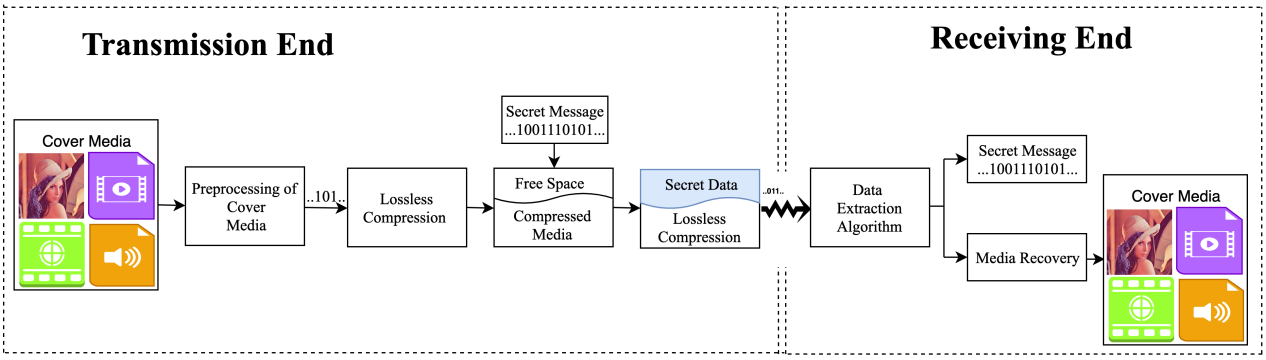

Fig. 2.4 General Framework of Lossless Compression Approach

cessed and then, lossless compression algorithm is applied on the preprocessed media for creating the free space. This free space is used for data embedding through an embedding algorithm. After data embedding, marked cover media is sent to the receiver. At the receiving end, the receiver extracts the secret information along with the cover media.

This approach is extensively investigated in the literature. For instance, in 1997, the first lossless compression method was introduced by Barton [1]. This method was applied for the authentication of cover media. It creates the free space for data embedding through lossless compression of cover media. But this approach was not proven experimentally. In a similar line, Honsinger et al.'s [30] proposed modulo based authentication method. For instance, secret information m was embedded into cover media M to generate marked media mM. Mark media $mM = (M+m) mod\ 256$, included hash value as embedding information. At the decoder end, embedded data m is retrieved from the marked media by subtracting the embedding data from the marked media, and the cover media is losslessly retrieved. However, marked media suffered from the salt and pepper noise during possible grayscale value flipping over the range of 0 to 255 because of modulo 256 addition. In order to overcome this issue, in [4], Fridrich et al. presented robust watermarking and lossless compression with encryption methods for invertible authentication. This robust watermarking approach overcame underflow/overflow problems due to addition modulo. Lossless compression was used by Joint Bi-level Image Experts Group (JBIG) method for compressing the bit planes of images. This method started observing the image from the 5^{th} bit-plane. In color images, all channels were compressed individually. If the redundant sum of all color channels was larger than or equal to 128, then 4^{th} LSB plane was considered for embedding. On the other hand, if the sum was less than 128, then 6^{th} bit plane was considered. This method was repeated until redundant sum is found greater than or equal to 128. Bit-plane provided enough space for hash embedding of cover media. In this order, Goljan et al. [4] introduced RDH based on regular-singular(R-S) method. In this, the image was partitioned into disjoint sets as Regular(R), Singular(S) and Unusable(U). The discrimination function was used to collect the smoothness of these sets. These blocks were assigned values as 1 for block R and 0 for block S, while U-block was unchanged.

Further, for each R and S-block, one payload bit was embedded either in R or S block. If the payload bit and the corresponding set do not match, then apply the flipping operation to the sets. The embedded data contains original payload along with overhead information. Decoder retrieved the bitstream from R and S blocks by scanning the image in the same direction as performed in the embedding process. The retrieved bitstream was separated into the payload and the compressed RS-vector stream. Original image was recovered by decompressing the RS compress vector stream. The real embedding capacity of this method was determine as, capacity $= N_R + N_S - |C|$, where N_R, and N_S were count of regular and singular block respectively, and $|C|$ is compressed bit-stream length. Results reveals that the estimated capacity lie between 15-30 bits according to the size of cover media. Maximum embedding capacity was achieved for groups $n \approx 4$ successive pixels, while flipping the seven operations with amplitude from 1 to 7 with PSNR of this scheme was 39-40 dB. This approach showed a small distortion after data embedding. Hence, this method has limited data embedding capacity.

Further, Xuan et al. [31] introduced integer discrete wavelet transform (IDWT) based upon lifting method [32]. In this, data embedding capacity of the method introduced by Goljan et al. [4] was improved. It was observed that the bias between 0's and 1's increased successively in higher bit planes up to some extent. This bias was used for creating free space by compressing the bits in bits plane. Secret information was hidden in this free space. In this approach, data was embedded only into middle and high frequency subbands of IDWT to improve the visual quality of marked media. Histogram shifting was used to prevent the overflow/underflow problem. Location map of the data was also embedded into the image as a overhead information. Average PSNR of this approach was 33.12 dB along with approximately 70028 bits embedding capacity for grayscale images. Later on, Celik et al. [3] introduced the generalization least significant bit(G-LSB) technique to enhance

the compression effectiveness by using arithmetic coding [33]. In this, the lowest level L of pixel values was decided by quantization. These levels were used for data embedding. Embedding capacity was enhanced as higher level pixel was considered. As the higher level was used, the distortion of the cover also increased. G-LSB adjusted the quantization track, and the data was embedded in the lowest levels of pixel values. This method was better than [2, 4] due to its scalability along the capacity-distortion. In [34], Chen et al. proposed RDH based on JPEG compression approach. In this, the cover image was compressed by JPEG compression technique. Further, a data hiding capacity table 8×8 was created according to a modified 8×8 quantization table. The capacity table and modified quantization table were used for all blocks of the cover image. After this, secret information was embedded into each quantized block. PSNR was 39.12 dB for a payload capacity of 253,952 bits with hiding ratio 35.23%. Further, He et al. [35] presented an improved block ordering frequency selection approach for JPEG images. In this, the distortion of each block was estimated using the discrete cosine transform coefficient distribution. Then, a block with less distortion was selected at first for data embedding. This method needs less side information for data extraction and cover media recovery when compared to some state-of-the-art methods.

Moreover, the RDH technique was suffering from the upper limit of payload embedding in cover media and the distortion constraint. This problem was solved by Kalker and Willems [36] for an independent identically distributed cover sequence. In [36], Kalker et al. constructed a specific rate-distortion problem in RDH, and achieved the rate-distortion function, i.e., the upper bound of the embedding rate for a given distortion constraint $\triangle$, where $\triangle$ is defined as in Eq 2.3.1:

$$\rho_{rev}(\triangle) = maximize\{H(S)\} - H(T) \tag{2.3.1}$$

where S, T are random variables of cover media and marked cover media respectively. The maximum entropy was over all transition probability matrices $P_{T|S}(t|s)$ satisfying the distortion constraint $\Sigma_{s,t} P_S PT|S(t|s)D(s,t) \leq \triangle$. The metric D(s,t) is called square error distortion, i.e., $D(s,t) = (s-t)^2$. Malik et al. [37] introduced an absolute moment block truncation coding (AMBTC) compression approach to achieve high capacity and good cover media quality. First cover image was compressed by AMBTC compression, and then divided into blocks of sizes $N \times N$. For each compress block, low mean and high mean were calculated. Difference between low mean and high mean decided either block was embeddable or non embeddable. Then, secret information was embedded into embeddable compressed block of cover image. For block size 4×4, this method achieved approximately 1.5 bpp capacity. Recently, in 2020, Lin et al. [40] proposed a two-layer embedding RDH approach based on $(7,4)$ Hamming code, in which, cover image was divided into non-overlapping blocks of size 4 and AMBTC compressed block. For the first layer embedding, 16 bits were embedded in each block, and 6 bits or 12 bits were embedded in each modified block in the second layer embedding using Hamming code. Embedding capacity obtained was 1.44 bpp along with PSNR 34.23 dB. Hence, this method achieved better embedding performance than the approach proposed by A. Malik et al. [37]. Table 2.1 summarises the performances of various Lossless compression RDH methods.

2.3.2 Correlation Expansion

Correlation Expansion based RDH approach uses redundancies in cover media for data hiding. Generally, these methods embed one bit of data into two pixels. Correlation expansion is classified as difference expansion and prediction error expansion. Difference expansion exploits the correlation of two adjacent pixels, whereas prediction error expansion exploits the correlation between neighbouring pixels. The details of these methods

Table 2.1: Recent Advances on Lossless Compression

Author	Methodology	Database	Result	Remarks
Barton[1]	Authentication based upon modulo 256	Self generated	Approximately less than 1.6% of image compressed using Lempel-Ziv compression approach	Error correction code was used for error correcting
Honsinger et al. [30]	Authentication based upon modulo 256 .	Self Generated	Embedding capacity improved of [1]	Modulo 256 avoid the overflow/under flow problem, watermark image can be suffer salt and pepper noise.
Goljan et al. [4]	Regular singular	Self Generated	The embedding capacity lies between 15-30 along with PSNR 39-40dB	Approximately group of four pixels provided better embedding capacity for all amplitudes.
Fridrich et al. [2]	Distortion-free data embedding techniques for the uncompressed BMP, TIFF and JPEG formats.	Self Generated	Average embedding capacity for different images of 0.0747 bpp along with PSNR 41.64 dB	Approximately group of four pixels provided better embedding capacity for all amplitudes
Xuan et al. [31]	Integer discrete wavelet transform (IDWT)	Self Generated	Approximately embedding capacity 70028-bits along with PSNR 33.12 dB	Limited data set used, histogram modification used to avoid the underflow/overflow problem
Celik et al. [3]	Generalized-least significant bit modification	Self generated	PSNR and embedding capacity depends on the embedding level as well as image's statistics	Outmatch Fridrich et al.'s [2] technique and embedding capacity depends upon quantization degree.
Chen et al. [34]	JPEG	Self Generated	PSNR 39.12 dB with capacity 2,53,952 bits	Data hiding ratio 35.23%
Zhang et al. [38]	Recursive code construction	CVG-UGR	Average payload capacity 0.03 bpp along with PSNR 62 dB	Improve method[36] by joint encoding and decoding.
Lin et al. [39]	Mean squared error, iterative algorithm	Self Generated	Embedding rate was 0.992 bpp, then mean squared error (MSE) was 7.278	Embedding rate and the distortion were increased together
Malik et al. [37]	AMBTC compression	Self Generated	Approximately 1.5 bpp for block size 4×4.	This method can be apply on color image also.
Lin et al.[40]	AMBTC compression, $(7,4)$ hamming code	USC-SIPI, Bossbase	Payload capacity 1.44 bpp along with PSNR 34.23 dB	Improved [37] payload capacity and visual quality.

are as follows.

2.3.3 Difference Expansion

Difference Expansion(DE) method exploits redundancy in the cover media. In this method, average and difference values of two neighbouring pixels are determined and, information bit is appended with this difference value. The marked cover media is generated by using the average and new difference value of neighbouring pixels. An overview of the difference expansion method is illustrated in Fig. 2.5. In 2003, first difference expansion approach was proposed by J. Tian [6]. In this, firstly average l and the difference values h were determined for two neighbouring pixels. Further, inverse integer transform was determined from pixel (x, y) with average l and difference h i.e, (l, h). The difference value h was split into four disjoint set as EZ, EN, CN, and NC. These disjoint sets are defined as $h \in EZ = \{0, 1\}$, $EN = h \notin EZ$, $CN = h \notin EZ \cup EN$, and NC which include all unchangeable h. Changeable difference values lie in the set $EZ \cup EN \cup CN$. Set EN was divided into two subsets $EN1$ and $EN2$. These sets decide the data embedding pixel. Those pixels were selected for data embedding for which the difference value h belongs to the set $EN1$. If pixel difference value $h \in EN2$, then this pixel will not contribute to data embedding. The embedding approach was applied multiple times to achieve an embedding capacity of more than 0.5 bit per pixel(bpp). However, this method suffered from embedding capacity and overhead information. The overhead information cost was almost double the amount of hiding information. To overcome this, the work was extended by Alattar [7] using a generalized integer difference transform. In this, pixels of a block of arbitrary size were considered instead of pixel pairs. A vector u of length n was formed from n pixel values. These values are chosen from n different locations within the same color component. Vector u was divided into three disjoint sets s_1, s_2, and s_3. These sets s_1, s_2 contain all expandable and changeable vectors

Table 2.2: Recent Advances on Difference Expansion

Author	Methodology	Database	Result	Remarks
Tian [6]	Difference Expansion, Haar wavelet transform	Self Generated	21675 message bits were embedded for block size 6×6 with the 100% accuracy of data extraction.	Two class SVM classifiers were used to discriminate encrypted and non-encrypted image patches
Alattar [7]	Generalized Integer Transform	Self Generated	Average embedding capacity 1.0 bpp along with PSNR 33.96 dB.	Improved Tian's method [6] using difference expansion of vectors instead of pairs
Kamstra and H. Heijmans et al. [26]	Least significant bit prediction, Sweldens' lifting	Self Generated	Payload capacity upto 0.5 bpp and for repeating the process achieve approximately 1 bpp.	Improved of Tian's [6] of difference expansion.
Lee et al. [41] et al.	Integer to integer wavelet transform, Le Gall 5/3 filters	USC-SIPI	Average payload capacity and PSNR, for grayscale 0.55 bpp, 36.90 dB, for color 1.15 bpp, 37.93 dB.	performance degraded for too small or too large block sizes.
Hu et al. [42]	Integer to integer wavelet transform	Self Generated	Embedding capacity approximately 0.5bpp	This approach outperforms [6, 26].
Quershi et al. [43]	Two-dimensional difference expansion, Standard deviation	Self Generated	Payload capacity between 0.81-0.998 bpp.	Achieved better hiding capacity than [6, 26].
Chen et al. [44]	Difference Expansion, Haar wavelet transform	Self Generated	For parameter k = 3 and Base = 40, PSNR = 16.8606 dB.	Generally, small block mean difference achieves better embedding capacity than conventional approach.
Arham et al. [45]	difference expansion of quad, multilayer embedding	idimages.org	For average payload capacity 0.007 bpp for common images and 0.018 for medical images.	PSNR increased much higher in layer 6 than the other one.

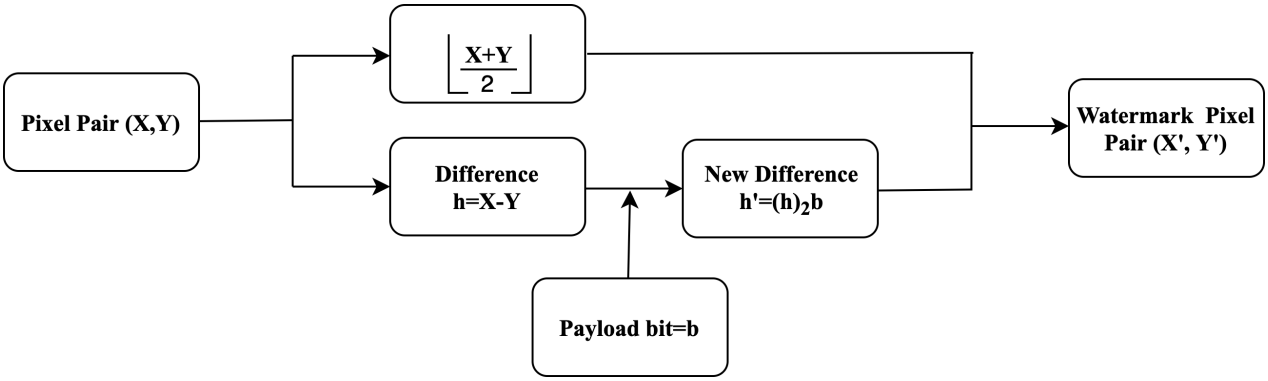

Fig. 2.5 General Framework of Difference Expansion

respectively, and the third set s_3 contains the rest of vectors-non changeable set. Vectors of s_1, s_2 and s_3 were identified using the location map M. Expandable vectors were used for data embedding and changeable vectors for the location map. The algorithm proposed by Altar is also able to embed 1.10 bpp with PSNR 37.94 dB. Embedding capacity depends on the image, and this approach worked only for color images. Following this, In [26], Kamstra et al. also enhanced the embedding capacity of Tian's method [6] using the LSB prediction embedding technique. In this, the cover image was converted into a binary bit stream, and then, the binary bit stream was compressed using an arithmetic coder. Secret information was embedded in the compressed media. At the receiver end, the original bit-stream and secret information were extracted without any error. Performance results of different difference expansion methods on different datasets are shown in the Table 2.2.

2.3.4 Prediction-Error Expansion

Prediction error expansion(PEE) attracted the researchers due to the spatial redundancy exploitation in the cover image. The correlation of two adjacent pixels was considered in the difference expansion-based RDH, whereas prediction error expansion methods use more neighboring pixels to exploit the correlation.

In the PEE approach, the data embedding process consists mainly of three phases.

Phase 1: Determine image pixels to get the sequence of prediction errors. Cover pixels work arranged in one dimensional sequence such as $(x_1, x_2, ..., x_N)$ prediction error of pixel x_i is determined as $p_{e_i} = x_i - \hat{x}_i$. Following this process, prediction error is generated.

Phase 2: Generate the prediction-error histogram.

Phase 3: Embed information bits by changing the prediction error histogram. For each p_{e_i}, it is changed as follows:

$$p'_{e_i} = \begin{cases} 2p_{e_i} + b, & \text{if } p_{e_i} \in [-C, C) \\ p_{e_i} + C, & \text{if } p_{e_i} \in [C, \infty) \\ p_{e_i} - C, & \text{if } p_{e_i} \in (-\infty, -C) \end{cases}$$

where C, b are payload capacity bound and secret information bit, respectively. At last, every pixel value x_i is changed into $x'_i = \hat{x}_i + p'_{e_i}$ to get the marked media. Maximum change in each pixel can be up to C, and the quality of marked media depends on the payload capacity bound C.

At the receiver end, first actual prediction-error p_{e_i} is retrieved from the marked prediction-error p'_{e_i} as follows:

$$p_{ei} = \begin{cases} \lfloor \frac{p'_{ei}}{2} \rfloor, & \text{if } p'_{e_i} \in [-2C, 2C) \\ p'e_i - L, & \text{if } p'_{e_i} \in [2C, +\infty) \\ p'_{e_i} + L, & \text{if } p'_{e_i} \in (-\infty, -2C) \end{cases}$$

At last original cover media was obtained by extraction of secret bits from LSBs of prediction errors $p'_{e_i} \in [-2C, 2C)$. Over the years, PEE works have been extensively investigated to achieve optimum performance. The details of recent work are as follows.

Prediction-error expansion approach was first introduced by Thodi et al. [27]. In this, the prediction error was determined for each pixel of the cover media using the predicted pixel.

For each pixel, the predictor pixel was calculated using three neighbour pixels. In order to create free space to hide secret information, the prediction error was shifted by one bit left, and the information bit was embedded in the LSB. To avoid the overflow/underflow problem, this method does not consider those pixels for data embedding, which has expandable prediction error beyond the set [0,255]. This approach outperforms the G-LSB algorithm proposed by Celik et al. [46] in the case of embedding rate more than 0.25 bpp. Later on, Thodi et al. [8] improved Tian's method [6] by using histogram and prediction error expansion techniques. At a low embedding capacity rate, distortion performance was enhanced by the histogram shifting method, and also reduced the capacity problem via the location map. PSNR of this method was greater than 50 dB for over datasets, while the PSNR of Tian's method [6] was always less than 50 dB. Embedding capacity based on prediction error expansion was 1 bpp with the same image quality, and it is double the value in comparison to the algorithm proposed by Tian's [6]. Further, Sachnev et al. [47] improved the embedding capacity of the method introduced by Thodi [8] by using the rhombus approach. In this, four neighbouring pixels were used for computing the predicted pixel, and all pixels of the image were divided into either a cross set or a dot set. The cross set was used only for data embedding, whereas the pixels of the dot set were used for pixel prediction. Prediction error was determined on the basis of predicted and original pixel values. This prediction error was expanded to hide the secret information, like the difference expansion method [6]. This method exploited sorting, histogram shifting, a double embedding approach, along with prediction errors with lower variance. Sorting rearranged the pixels according to the magnitudes of local variance. Embedding capacity was one bit per pixel of this method using double embedding. In [48], Caltuc proposed a prediction expansion reversible watermarking approach based on median edge detector(MED), gradient adjusted predictor(GAP), and simplified GAP predictors. This ap-

proach used USC-SIPI and Kodak datasets for the experiment. USC-SIPI dataset includes grayscale images of size 512×512, whereas Kodak includes color images of size 512×768. Experiments were performed with different predictors, and this approach improved methods proposed by Thodi [8, 27]. Average gain of PSNR was 0.84 dB using MED predictor with location map and 0.85 dB gain by using histogram shifting for MED predictor over USC-SIPI dataset [49]. In a similar line, Uyyala and Pal [50] discussed an improved gradient approach based on prediction, in which the gradient was calculated on the basis of four directions by using 5×5 neighbourhood pixels. In [51], S. Yang introduced a saliency fixation map based on a saliency prediction method to enhance the visual quality of cover media. However, these methods did not consider adaptive embedding of secret data.

In order to solve this, Li et al. [52] proposed an adaptive embedding and pixel selection approach based on prediction error expansion. This approach combines PEE, adaptive embedding, and pixel selection techniques. In this, the image was divided into smooth and rough regions, and then two bits of information were embedded into the smooth region and one bit of information was embedded into the rough region. Performance of this approach was improved over [28], wherein an average embedding rate of 0.098 was obtained. However, an adaptive embedding threshold was used for image partition, but this was determined iteratively. Further, in [53], D. Coltuc proposed prediction error expansion based on a transform by modifying the four pixels. A change in four pixels guarantees less distortion than a change in one pixel. This approach introduces low distortion by using a median-edge detector (MED) and gradient adjusted predictor (GAP). The average embedding capacity of this approach was 0.34 bpp along with a, PSNR of 70 dB. Additionally, PEE has been investigated for other correlation among pixels of cover media.

Further, Li et al. [54] proposed PEE based on pixel value ordering (PVO). In this approach, the cover image was divided into non-overlapping blocks of the same size, and pixels of

each block were arranged in ascending order. The largest pixel value of each block was predicted by the second largest pixel value, and the corresponding prediction error was calculated as $p_e = p_n - p_{n-1}$. Information bit was embedded only with the largest pixel value of each block. The maximum change of pixel value was one after embedding the information bit. PSNR performance was increased. The embedding capacity of this approach depended on the block size. The average PSNR of this method was 58.56 with an embedding capacity of 10,000 bits. In a similar line, Peng et al. [55] improved the method by Li et al [54] by exploiting the blocks, where the largest and second largest pixels were equal. PSNR was improved by 0.42 dB for an average embedding capacity of 10,000 bits over the USC-SIPI database. Experiments were also performed on the Kodak and BOSS-Base databases. Average PSNR for these databases improved over X. Li et al. [54] by 1.64 dB and 0.69 respectively, for 10,000 bits. Further, in [56], Jain and Kasana extended the RDH approach proposed by Peng et al. [55] by determining the difference between the largest and second largest pixel values locations. Negative difference values were used for creating free space for data hiding. Average PSNR was 58.25 dB, corresponding to 20,000 bits, and provided better PSNR than proposed methods [47, 54, 55].

In [57], Kumar and Agarwal presented fragile RDH using PEE, in which odd columns were used to predict the even columns' values. Secret information was embedded into even columns. Embedding capacity for cover image of size $m \times n$ was $(m \times (n/2-1))$ with average PSNR of 34.66 dB. In [58], Ou et al. presented a reversible data hiding approach based on prediction error expansion. This approach exploits the correlation among the prediction errors. In this, the cover image is divided into shadow and blank sets.

The first line pixels of the shadow and blank portion were excluded because these pixels were used to embed some parameters for the purpose of data extraction. The sequence of shadow pixels is determined according to the defined order. Then, rhombus predic-

Table 2.3: Recent Advances on Prediction Error Expansion

Author	Methodology	Database	Result	Remarks
Thodi et. al [8]	Prediction error expansion, histogram shifting	USC-SIPI	Capacity upto 1 bpp with PSNR 50dB	Improved embedding capacity of [6] Algorithm.
V. Sachnev [47]	Rhombus pattern prediction	Self Generated	Maximum embedding capacity 1 bpp	Decreased the size of location map
D. Colutic [48]	MED, GAP, SGAP	USC-SIPI, Kodak	Improved data hiding capacity of [8]	Low data hiding capacity
Li et al. [52]	Adaptive embedding, pixel selection	Self Generated	Increased average embedding rate 0.098 bpp compare to [28]	Data embedding threshold iteratively determine
D. Colutic [53]	PEE transform	Self Generated	Average embedding capacity 0.34 bpp along with PSNR 70 dB	Achieve low distortion using MED and GAP predictor
Li et al. [54]	Pixel value ordering	Self generated	PSNR 58.56 along with 10,000 bits.	Change of pixel after one bit embedding was one.
Peng et al. [55]	Pixel value ordering	USC-SIPI, Kodak, Bossbase	Average embedding capacity increased 14% compared to [54], PSNR improved 0.42 dB compared to [54] using USC-SIPI.	Experiment performed on Kodak and Bossbase databases.
Ou et al. [58]	Correlation between prediction error	USC-SIPI, Kodak, BOSSbase	Approximately PSNR 55 dB with capacity 20,000 bits.	Different PSNR for different databases with the same embedding capacity
Ou et al. [59]	Pixel value ordering	Self Generated	Average PSNR 59.17 dB along with 10,000 bits, 55.47 dB along with 20,000 bits.	Extended the work of X. Li [52].
Abbasi et al.[60]	Generalized PVO, firefly quadtree partition	USC-SIPI	Average embedding rate 36,333 bits along with PSNR 51.66 dB.	Improved embedding rate and visual quality compare to methods [54, 55, 59]

tion [52] is computed for shadow pixels. After generation of the prediction error sequence $(e_1, e_2, ..., e_N)$, it was divided into pairs such as $E_i = (e_{2i}, e_{2i-1})$ where the prediction error pairs are arranged in decreasing order of prediction error accuracy. Prediction error accuracy was measured by the local complexity. Those pairs that have local complexity less than or equal to some threshold, bound were used for data embedding, and the others were skipped. A similar approach was applied to the blank set of pixels. This approach enhances the rhombus prediction [52] by using local gradients, and it was also better than [47, 52] upto some extent by using GAP for prediction. Three databases namely: USC-SIPI [49], Kodak [61], and BOSSBase [62], were used for the experiment. For 10,000 bits, the average PSNR is 59.08 dB, while for 20,000 bits, the PSNR is 55.00 dB over the USC-SIPI database [49]. Another experiment was performed over the Kodak image database, which contains color images. For experimental purposes, Color images were converted into grayscale images. In comparison to [47], average PSNR improvements were 1.34 and 1.50 dB for a capacity of 10,000 and 20,0000 bits. By using the BOSSbase database, average PSNR improvements were 1.33 and 2.24 dB in comparison of the methods [47] and [52], respectively. In this sequence, Ou et al.[59] extended the Li et al. work [52] wherein the cover image was divided into non-overlapping blocks of size $m = m_1 \times m_2$. First, pixels of each block were arranged in an ascending order. The maximum pixel p_m was predicted by the pixel p_{m-1}, and the derived prediction error x was determined as $x = p_m - p_{m-1}$. Information bit was embedded as the maximum pixel value of each pixel, and this pixel will be either unchanged or have its value increased by one while other pixels remain unchanged. At the receiver end, the receiver determines the marked prediction error as $x' = p'_m - p'_{m-1}$. Information bits extracted as 0 and 1 as the marked prediction error x' is 1 and 2. In [60], Abbasi et al. introduced RDH based on generalized PVO using the firefly method. In this, the cover image was partitioned into non synchronized dynamic blocks according to

the firefly quad-tree method for compressing the cover image. Sequence blocks for the minimum and maximum number of pixels were calculated. A block of minimum number of pixels was used for prediction and modification, while secret information was embedded into blocks of maximum number of pixels. Average embedding rate and PSNR were 36,333 bits and 51.66 dB, respectively. Performance was better than the proposed methods [54, 55, 59]. However, most of these approaches do not improve security and privacy concerns.

In order to address security concerns, in [63], Kumar and Jung introduced robust RDH to avoid non-malicious attacks by using two layer embedding. In this, the cover image was divided into higher significant bits and lower significant bits planes. Then, secret information was embedded into higher significant bit planes by using prediction error expansion. Later on, in [64], Li and Huang presented a JPEG RDH approach based on pairwise nonzero Alternating Current (AC) coefficient expansion to enhance the visual quality of the cover image. In this, a two-dimensional histogram was constructed of non-zero AC coefficient pairs, and then secret information was concealed based on modified non-zero AC coefficients. Performance results of the prediction error expansion methods on different datasets are shown in the Table 2.3.

2.3.5 Histogram Shifting

A histogram is a graphical representation of data points. This organizes a group of data points into a specific range defined by the user. It looks similar to a bar graph. The maximum and minimum number of occurrences of elements in the set of data points are called peak point P and zero point, respectively. Histogram shifting is an effective method for reversible data hiding. Various authors introduced histogram based reversible data hiding approach in the literature. In 2006, Ni et al. [9] proposed the first Histogram shifting

method. In this technique, the sender generated a histogram of the image and selected the peak and zero points for data embedding. If the message bit 1 is encountered, then the histogram is shifted to the right hand side by 1, otherwise, no modification is required. The capacity of embedded data bits in image is equal to the number of peak values. It embeds 5-80 kb data into a $512\times512\times8$ grayscale image with a PSNR value of more than 48.13 dB of marked image was reported. For a $M\times N$ image, the complexity was $O(3MN)$. This approach suffered from the issues of multiple zero points, which required the storage of the coordinates of zero points. Further, Fallahpour and Sedaaghi [65] improved the embedding capacity and PSNR value of Ni et al. method. In this technique, HS was applied to a block instead of the whole image. The image is divided into n blocks, and then each block's HS is generated. Determine peak point P_i and zero point Z_i for the i^{th} block. For $P_i > Z_i$ then all pixel values belong to $[Z_i+1, P_i]$ and the histogram is shifted to the left by 1. Pixels P_i-1 increase by one, if the embedding data bit is 1, otherwise not to be altered. For $P_i < Z_i$, the greyscale pixel values will be from P_i+1, to Z_i-1, and the histogram shifted to the right by 1. Pixels P_i increase by one if the embedding data bit is 1, otherwise, no change is required. Embedding capacity for greyscale Lena and Baboon image $(512\times512\times8)$ are approximately 154% and 46% more than [9]. This technique endure zero points and peak points information that needs to be transmitted to the receiving side for data retrieval. Furthermore, Lee et al.'s [11], proposed a method based on the histogram of difference image. This approach outshines Ni et al.'s [9] by improving both embedding capacity and visual quality. The spatial correlation of natural images is exploited in Lee et al.'s [11] scheme. For instance, a grayscale image $I(i,j)$ of size $M\times N$ the difference image $D(i,j)=I(i,2j+1)-I(i,2j)$, where $I(i,2j+1)$ and $I(i,2j)$ are called odd line pixel and even line pixel, respectively. For $D(i,j)\geq 2$, and $D(i,j)\leq -2$,, add 1 and subtract 1 from the odd line pixel step by step. Otherwise, no modification

required. Modified difference image is used for data embedding. If the embedded bit is 1, move the difference value of -1 to -2, subtract 1 from the odd line pixel, or 1 to 2, adding 1 to the odd line pixel. If the embedded bit is 0, skip the pixels of the difference image until a pixel with a difference value of -1 or 1 is encountered.

Lin et al. [66] proposed a multilevel reversible data hiding method. In this method, the image is divided into non-overlapping blocks. Calculate the difference between the adjacent pixels and record the peak point for each block. If the pixel value of a particular block is larger than the peak point P of the same block, increase the pixel value by 1. Otherwise, the pixel value remains unchanged. If the difference value of a pixel is the same as the peak point of the block, then add the message bit to the difference pixel, using the original image and its difference image to construct the marked image. At the receiver end, divide the received marked image into blocks and generate its difference image. If the difference is equal to a peak point P and $P+1$, then the embedded bits are 0 and 1, are retrieved respectively. At level 1, this hiding algorithm can hide 0.25 bpp. This method embeds 2 bits per pixel for using 18-level hiding, the embedding capacity decreases if the number of levels roughly increases. Computational complexity of k block with block size AB is $O(5ABk)$. The lower bound of PSNR was about 42.69 dB. In [67], Tai et al.'s proposed histogram modification method is based on the absolute values of the adjacent pixel difference. In this method, the binary tree structure was used to solve the issue [9, 65] of multiple pairs of peak points for communication. Also, to control the overflow and underflow problem, the HS scheme can be applied to narrow the histogram from both sides. Only the tree level information was shared between the sender and recipient with the payload. However, pure payload unexpectedly decreases with the increase in the tree level, and overhead information increases with the tree level. In [68], a common structure for designing histogram based RDH was introduced, and the present structure incorporates numerous past

techniques. For this common framework, the cover media was firstly partitioned into non-overlapping sets, and each set included n pixels, and by counting the frequency of each set the n-dimensional histogram was generated. Then, data embedding was finished by emending the n-dimensional histogram. Every pixel block was partitioned into two disjoint sets, in which one set was used for hiding the secret data and the other one was used for creating free space by shifting pixels and confirming the reversibility. However, this framework may design different RDH techniques, but it also has some limitations. A few Histogram Shifting based techniques namely location map-free [69, 70] and adaptive [52] schemes, can not be constructed by this framework. In [71], Ou et al. proposed the RDH method based on multiple histogram modifications to achieve high embedding capacity by using multiple pairs of bins for every prediction error histogram. Maximum embedding rate was 0.4 bpp along with average PSNR 48.50 dB [47, 72]. In this sequence, Qi et al. [73] proposed RDH based on multiple histogram modifications to achieve high payload capacity. Multiple pairs of bins were used for each histogram to increase the hiding capacity. The average PSNR was 59.82 dB, corresponding to 10,000 bits. In [74], Garg et al. introduced RDH based on the histogram and modulus operator. In this, the cover image was divided into blocks and modified these blocks to increase the histogram peak points. Then, secret information was embedded into blocks using peak points. Performance was better than proposed methods [9, 47, 65, 66] with respect to payload capacity and PSNR. In [77], He et al. discussed RDH for high dynamic range(HDR) images incorporating new luminance channel. In this, edge information was used among luminance and color channels to accomplish correct prediction and high data hiding capacity. Then, secret information was concealed in HDR images, and to reduce visual loss, an edge directed order was also introduced. In this sequence, Gao et al. [78] suggested RDH based on the prediction error histogram for HDR images. In this, the first prediction -errors were determined through

Table 2.4: Recent Advances on Histogram Shifting

Author	Methodology	Database	Result	Remarks
Fallahpour and Sedaaghi [65]	Relocation of zeros, peak of histogram	Self generated	Embed approximately 5k-250k bits into 512×512 grayscale image	Only one method was used for comparison
Lee et al.[11]	Authenticationon based on watermark image	Self generated	PSNR higher than 51.14, capacity lie between 8150 to 28094 bits for grayscale images.	Hash function used for integrity, trade-off between capacity and distortion
Lin et al. [66]	Multilevel difference image histogram modification	Self generated	Average PSNR 30dB and capacity 1.3bpp.	Measure joint imperceptibility and hiding capacity
Tai et al. [67]	Histogram modification	self generated	Average PSNR higher than 30 dB and capacity 0.0985 bpp	Only six grayscale images were used for experiment
Gao et al. [75]	Generalized statistical quantity histogram	CVG-UGR, Brodatz, DICOM, OsiriX website	Capacity depends on block size and scale factor	divide-and-conquer strategy was used to solve the overflow and underflow
Li et. al [68]	Nine-dimensional histogram, shifting function, embedding function	USC-SIPI	Average PSNR 39.83 dB corresponding to 0.5 bpp embedding capacity .	One RDH framework and two RDH approaches proposed
Li et. al [76]	Difference pair mapping	USC-SIPI	Average PSNR 55.67 corresponding to data hiding capacity 0.076 bpp .	Pixels 0 and 255 may be participated in Overflow/underflow due to modification at most one
Garg et al. [74]	Histogram shifting and modulus operator	Self Generated	Improved payload capacity and PSNR of methods [9, 47, 65, 66].	Small block size provided higher payload capacity
Li et al. [72]	Multiple histogram modification	USC-SIPI	Average PSNR 59.74 along with 10,000 bits payload capacity.	Improved PSNR 2.79 dB compare to the method [47].

encoding of HDR images, while complex region pixels determine the second prediction errors. Then, two dimensional prediction-error histogram was generated to optimize the data hiding capacity. Further, an adaptive embedding approach based on two dimensional prediction-error histogram was adopted to overcome the image distortion due to data hiding. Average PSNR was 60.01 dB, corresponding to 15,000 bits and outperformed the proposed state-of-the-art methods [47, 77, 79]. In [79], et al. introduced RDH based on image texture to overcome unreasonable shifting of pixels. In this approach, the original image was split into two segments using a cheeseboard pattern, and then fluctuation values for each segment were computed. The next secret information was embedded into the low fluctuation region of the segmented cover image.

Wang and Wang [80] presented histogram based RDH approach by using prediction error. In this, the cover image was partitioned into non-overlapping blocks, and then the median of each block was computed. In the embedding phase, pixels were expanded into negative and positive directions according to the median value, and the median pixel remains the same after data hiding. The decomposed location message was used to control the overflow/underflow problem by modifying the original pixels. In [81], J. He et al. proposed an RDH approach based on negative influence models for data embedding in JPEG images. In this, a negative index for each frequency was described with a weighting factor. This weighting factor was adjusted by the user's choice for low distortion. Priority of small negative indices frequencies was given for data embedding. Secret information embedded into quantized AC coefficients except zero according to the zero run length method in ascending order, and the overhead information was $69+\log_2(64N)$ bits. Table 2.4 summarizes the Performance results of the histogram shifting methods on different datasets.

2.3.6 Interpolation

Interpolation is a method of estimating unknown values between known values.

In 2009, Jung and Yoo [15] proposed a neighbour mean interpolation(NMI) based data hiding approach. In this, the first cover image was scaled up using neighbour mean interpolation, and then, the difference value d was determined for four non-overlapping consecutive pixel values. Further, each scale-up pixel was embedded with $\lfloor log_2|d| \rfloor$ bits. Embedding capacity of this method was about 112 - 400 KB, along with PSNR more than 35 dB. This had given better payload capacity than other scale-down method. Yalman et al. [82] proposed the RDH method based on the NMI approach by using R-weighted Coding. Embedding capacity was 1.91 times better than proposed by Jung [15] method, along with PSNR 39 dB.

In [83], Chang et al. proposed two level information hiding approach based on interpolation and histogram schemes. In this, the cover image C was generated corresponding to the input image I by using the enhanced NMI technique. Further, the difference image d was computed as the difference of the cover image C and the input image I. Then, the binary sequence of secret information was divided into a sub-sequence before hiding. In this subsequence, bit "1" was padded with MSB on the left-hand side. This new subsequence was converted into decimal digits d, and the first level of data hiding was done by embedding d into pixels of the cover image. Stego image S_1 was generated, and to increase the payload capacity, a second level of data hiding was done through histogram modification. A new difference image was computed with the difference of the stego image S_1 and the cover image C, and the histogram of this difference image was created. Then, two peak points were chosen, and the first histogram was shifted according to the peak points and the cover image pixel. After this, secret information was embedded. The average

Table 2.5: Recent Advances on Interpolation

Author	Methodology	Database	Result	Remarks
Jung and Yoo [15]	Neighbour Mean	Self Generated	Payload capacity between upto $400KB$ along with PSNR 35 dB.	Efficient and low time complexity, four image used for experiment
Lee and Huang [84]	Interpolation by neighbouring pixel	Self Generated	Average payload capacity 1.56 bpp along with 21.82 dB.	Improved payload capacity of methods [15] from 56% to 108% along with same PSNR value
Chang et al. [83]	Enhance neighbour mean interpolation, Histogram modification	Self Generated	Average payload capacity 1.04 bpp along with PSNR 41.90 dB	Improved payload capacity as well as PSNR of method [15]
Jana and Jana [85]	Interpolation	Self Generated	Payload capacity from 2,57,612 to 7,75,832 bits, PSNR from 36.71 dB to 42.82 dB.	Sixteen grayscale image of size 256×256 used for experiments.
Tsai et al. [86]	Reference pixel mapping	Self Generated	Embedding capacity lie between 1.26 to 2.58 bpp	Improved payload capacity from 94.75% to 198.87% compare to the [15] method
Malik et al. [87]	extended neighbour mean interpolation	Self Generated	Improved payload capacity 9.50% and PSNR 19.45% compare to method [88].	Improved visual quality of interpolated image
Hu and Li [89]	Maximimizing Difference value between Neighbouring pixels	USC-SIPI	Average capacity 1.81 bpp and PSNR 33.03 dB	Four grayscale image used for experiment.

embedding capacity was 1.04 bpp, along with PSNR 41.90 dB. This method gave better performance than [15] method. Zhang et al. [90] introduced RDH based on parabolic interpolation, which is used for medical applications. First, interpolated image $P = (p(i,j))_{2m\times 2n}$ was generated by rescaling the input cover image $Q = (q(i,j))_{m\times n}$ by factor two. In this, interpolated image P, $p(i,j) = q(i/2, j/2)$, if i and j were even integers, otherwise $p(i,j)$ were determined by parabolic equation $y = ax^2 + bx + c$. In the data embedding phase, this method unchanged those pixels, where $p(i,j) = q(i/2, j/2)$ was unchanged, and secret information was embedded by exploiting the interpolated pixels. On payload capacity 1.09 bpp with PSNR 28.41 dB, while Jung and Yoo [15] have PSNR 27.31 dB. Shaik et al. [88] introduced another high capacity RDH 2D parabolic interpolation. Malik et al. [91] proposed an interpolation approach for RDH based on pixel value adjusting feature. In this, firstly, the input image was interpolated based on the NMI method using different weightages. Then embedding was done in two phases. In the first phase, odd valued pixels were used to hide secret information, while in the second phase, secret information was embedded into even valued pixels. PSNR was increased from 4.90% to 29.60% when compared to the method proposed by Chang et al.[83].

In 2020, Jana and Jana [85] introduced a method to enhance payload capacity. In this, two interpolated pixels were included in two pivot pixels. These interpolated pixels were determined in such a way that the values of these pixels were close to pivot pixels. Then, a substring of secret information was embedded into two interpolated pixels. Embedding capacity and PSNR varied from 2, 57,612 to 7,75,832 bits and 37.71 to 42.82 dB respectively. This capacity and PSNR were better than previous proposed methods [15, 83, 84, 89]. Further, Malik et al. [87] discussed RDH based on pixel intensity for interpolated images. First, pixels were divided into three groups with intensity range $(0-15, 192-255)$, $(16-31)$ and $(32-191)$. Then, interpolated pixels were identified for embedding the secret information.

If the identified pixel was contained in the first intensity range group, then 4-LSB of the identified pixel was replaced by 4 bits of secret information. If this identified pixel was contained in the second or third group of pixel intensity, then 3-LSB or 2-LSB was replaced by 3 or 2 secret information bit in sequence. The average gain of embedding capacity of this method was 9.50% with PSNR 19.45% than the previous method [85]. In 2012, Lee and Huang [84] improved the Jung and Yoo method [15] by using an interpolation by neighbouring pixels approach. This approach uses an overlapping block of size 3×3 of the cover image to enhance embedding capacity. The maximum value M was identified among the four corners for each block. Three different values were calculated using three neighbouring pixels of the pivot pixel from the maximum value M. Further, $\lfloor log_2|d_i| \rfloor$ bits were embedded for each neighbouring pixel corresponding to pivot elements. Embedding capacity was increased from 56% to 108% than the Jung and Yoo method [15] at the same PSNR value. Tsai et al. [86] improved embedding capacity and visual quality by proposing Lee and Huang's [86] method by modifying reference pixel calculation and pixel adjustment procedure. Payload capacity was improved from 94.75% to 198.87%. Hu and Li [89] introduced reversible data hiding based on extended interpolation to achieve high embedding capacity and visual quality. In this, the difference was maximising amid neighbouring pixels to increase payload capacity from 14.5% to 35.5%, in comparison with the neighbouring mean interpolation method. The average payload capacity of this scheme was 1.81 bpp with PSNR 33.03 dB. Table 2.5 summarises the Performance results of the interpolation methods on different datasets.

2.4 Encrypted Domain

In the last few years, the challenge in front of the scientific community is to combine compression, encryption and data hiding all together. It is well known that encryption is an effective technique of privacy protection for securely sharing information with others. The information owner can transmit the media after encryption. In some scenarios, a database administrator appends some additional data (ex., origin of information, authentication data) within the encrypted media without knowing the original media. For example, a database administrator may embed the patient's personal information into the encrypted images to protect the patient's privacy. At the receiver side, the original data can be recovered after decryption without any error and also receive the additional message. On seeing such types of scenarios reversible data hiding (RDH) algorithm are required for the encrypted domain (RDH-ED). The purpose of RDH-ED is to embed additional data into encrypted media without revealing the plaintext data, and at the receiver side, error-free original plaintext data is obtained. Reversible data hiding in encrypted domain (RDH-ED) algorithm is classified into two categories, either vacating room after encryption (VRAE) or vacating room before encryption(VRBE). In the next section, both methods have been reviewed.

2.4.1 Reserving Room after Encryption

In the Reserving Room after Encryption (RRAE) approach, the original cover media is encrypted without any preprocessing by the content owner. Further, the data hider embeds the secret information into encrypted cover media by modifying some bits. The embedding capacity of this approach is limited due to the high entropy of the encrypted image. Also, Secret information extraction and cover media recovery are not satisfactory. But the op-

erations of this approach at the end user are simple and efficient. RRAE work can be categorised as joint and separable RDH. The details of these approaches and frameworks are as follows, and their performance results on different datasets are summarised on Table 2.6.

Joint

In joint schemes, with an encrypted cover media containing secret data, a receiver may first decrypt it with the decryption key, and then retrieve the secret information and original cover media recovery from the decrypted cover media using the data-hiding key. In 2008, the first joint vacating room after encryption (J-VRAE) approach was proposed by Puech et al. [92]. In this, the original image is encrypted by applying the Advanced Encryption Standard(AES), then divided into blocks with 16-pixels of each block and embedded one bit of data in each block. At the receiver end, data extraction and image recovery were obtained by examining the regional standard deviation. The embedding capacity of this method was 1 bit for each block of 16 pixels. In this sequence, Zhang [23] proposed RDH in the encrypted domain based on LSB modification. In this, the original image was encrypted using the encryption key. Then, the data hider partitioned the encrypted image into the disjoint blocks of size $S \times S$. Each block was divided pseudo-randomly into two sets s_1 and s_2. At the stage of data embedding, the three least significant bits of each pixel to the set s_1 and s_2 were flipped corresponding to the information bits 0 and 1, respectively. At the receiving side, the receiver decrypts the encrypted image using the decryption key. The first five MSB of each pixel are the same as the original image, but the three LSB of each pixel are different from the original one of the decrypted image. Further decomposed the decrypted image into blocks and pixels of each block divided into two different sets s_1 and s_2 as in the embedding phase. Flip the three LSBs of each pixel of these sets and

form new blocks h_1 and h_2. In these new blocks, either h_1 or h_2 was the actual block, and another one was tampered with because of the LSB flip operation. For the data extraction and image recovery, the receiver calculates the fluctuation values f_1, f_2 corresponding to the blocks h_1 and h_2. If the fluctuation value $f_1 < f_2$, then the block h_1 was originally extracted and $a's$ bit is 0. Otherwise, block h_2 was original and extracted bit be 1. This method can embed a 256-bit into a grayscale encrypted image along with a PSNR of 37.9 dB. More data can be embedded for the small block size, but the error of bit extraction and image recovery can be increased. If block length is not less than 32, then it can extract the embedded bit as well as the original image without any error. This method suffered from two issues: one, it did not properly utilise the pixel smoothness of each block and the other, pixel correlations in the boundary of vicinal blocks were also not used. In [93], Hong et al. considered these two issues and improved Zhang's method [23]. In order to reduce the error rate of extracted bits, the pixel smoothness of each block was used. In this, the side match method was adopted to check the smoothness of each block. The error rate of this method was 0.34%, while the error rate of Zhang's method [23] was 1.21%. In this order, Yu et al. [25] also improved Zhang's [23] method by modifying the fluctuation function. This reduced the error rate monotonically. Liao and pixels [94] improved methods Zhang's and Hong et al. approaches [23, 93] based on block complexity. In this, a complexity function was proposed to determine the complexity of each block of the image. This decreased the average error of image recovery and the extracted bit error rate. For the block size greater than 8, the extracted bit error rate was lower than the methods [23, 93]. Extracted bit rate was 0.66%, whereas Zhang's and Hong et al. approaches were 1.32 bpp and 1.32 bpp, for block sizes 32. In [95], Di presented bit-plane operations and an adaptive embedding approach for RDH in the encrypted domain. In this, the encrypted cover image was divided into two parts according to bit-plane operations, and an adaptive strategy was

used for data hiding. Embedding capacity was better than proposed by the Huang et al. [96] method.

Later, Wu et al. [97] proposed joint and separable methods for the encrypted domain. Firstly, the original grayscale image of size was encrypted by the encryption key. In the joint approach, the encrypted image was divided into qualified and non-qualified disjoint sets. A qualified set was adopted for data embedding, and locations were chosen randomly according to the data hiding key. If the randomly selected location did not belong to the non-qualified set, then these locations were considered as into the qualified set until the predefined threshold was exceeded. The data embedding rate of this method was 0.0625 bpp, and the original image was recovered without any error. The other method was separable. In this, secret information bits were embedded using MSB substitution. On the receiver side, a median filter was used to recover the secret information from the marked cover image. The embedding capacity of this approach was at most 0.1563 bpp. In [98], Zhang proposed a homomorphic and block encryption based RDH approach. In this, the cover image was divided into non-overlapping blocks, and then encrypted using homomorphic and block permutation encryption. Further, the encrypted cover image was preprocessed using prediction methods. This encrypted cover was partitioned into three sets, namely set 1, set 2, and set 3, according to sub-blocks locations. For maximum use of redundant sub-blocks, various prediction methods were used for these three sets. These three sets were used median edge detector predictor, the rhombus predictor and the mean of vertical and horizontal directions pixels, respectively. Two layers data embedding was done by adopting a histogram shifting approach based on the calculated prediction error. At the receiver side, the approximate cover image was extracted by the decryption key, and the data was extracted according to the embedding methods in reverse order. Six images for the experiment from the public domain dataset CVG-UGR were used. Embedding ca-

pacity was 0.5 bpp along with PSNR 41.95 dB, and this approach was superior to [19] with 6.86% dB PSNR in comparison to [19, 23, 99] methods. Recently, in 2020, Kumareson and Gopalan [100] proposed RDH based on cellular automata for the encrypted domain to enhance payload capacity along with low complexity. The receiver recovered the cover media without any error with higher PSNR, when embedding capacity was less than 1024 bytes.

Separable

In a separable, the content owner encrypts the cover media using the encryption key and sends the encrypted media to the data hider. Secret information is embedded into encrypted media by a data hider using a data hiding key. At the receiving end, the receiver can extract secret information or decrypt the cover media depending on available keys. Zhang's [24] proposed the first separable RDH in an encrypted domain approach for RRAE. In this technique, the original image was encrypted using the encryption key generated by a stream cipher. Then, the encrypted image was pseudo-randomly permuted and divided into blocks of size L. T LSB-bits of each block are compressed with a parity-check matrix, and data is embedded into the vacated space. For instance, pixels of one block denoted by $y_1, y_2, ...y_L$ and its encrypted T LSB-bits by q that consist of $T.L$ bits. The data hider generates a parity-check matrix G of size $(T.L-S)\times T.L$, and compresses as its syndrome s such that $s = G.q$. At the receiver side, the $8-T$ most significant bits (MSB) of pixels are obtained by decryption directly. The receiver estimates y_i $(1 \leq i \leq L)$ by the MSBs of neighbouring pixels, and gets an estimated version denoted by q and q' . On the other hand, the receiver tests each vector belonging to the coset $C(s)$ of syndrome s, where $C(s) = \{v|G.v = s\}$. From each vector of $C(s)$, the receiver can get a restored version of q , and select the one most similar to the estimated version q' as the restored LSBs.

Complexity of this method for data recovery was $\bigcirc(N, 2^s)$, where N size of the original image, and the embedding rate of the encrypted image with embedded data was 0.017 bits per pixel(bpp). PSNR of the directly decrypted image was 39.0 dB and a gain of 8.7 dB of PSNR when compared to [23] method.

Further, Zhang et al. [101] proposed RDH for an encrypted domain based pseudorandom sequence modulation process. then, the cover image was encrypted using an encryption key and divided into t non-overlapping blocks of size $m \times m$. 3-LSB plane for each block was pseudorandomly permuted. Then, the data hider was embedded B bits into each block. PSNR of this scheme was 37.9 dB and better than [23] with 13.3 dB, and the data embedding rate depended on the block size. For small block size and large embedding data, leading to the risk of wrong data extraction and original recovery. Embedding rate achieved was 0.0031 bpp for block size 40×40 with error free data extraction.

Further, Chen et al. [102] proposed reversible data hiding in the encrypted domain based on a public key cryptosystem and homomorphic encryption approach. In this, each pixel was divided into two parts, i.e, 7 MSB and one LSB, and then, these were encrypted respectively. Homomorphism properties were adopted to embed one secret message bit reversibly into two modified LSBs of each encrypted pixel pair. At the receiver end, embedded secret message bits were extracted by determining the relationship between LSBs of two decrypted pixel pairs. Average PSNR was 42.85 dB along with 0.25 bpp embedding rate. This approach suffered from the inherent overflow. In [103], the cover image was encrypted by a chaotic sequence. Then, encrypted cover media was partitioned into non-overlapping blocks. LSBs of pixels from each block were extracted to generate the LSB sequence, and the corresponding auxiliary sequence was also generated. The Hamming distance between these two sequences was calculated. Then, LSBs of the encrypted image were compressed using Hamming distance, and space was created for data em-

bedding. The average embedding rate was 0.041 bpp, along with a PSNR of 55.0 dB of the marked decrypted image. Yin et al. [104] proposed a multi-level encryption approach based on multi-granular and stream cipher, and also presented block histogram modification approaches. In this, the cover image was divided into non-overlapping blocks. These blocks were randomly permuted using the Josephus traversing method. Then, permuted blocks were encrypted by the multi-granular encryption approach and generated an encrypted image further encrypted using the stream cipher. Authors overcome the classical histogram modification problems, such as side information, overflow/underflow, using a block histogram modification approach. This method embedded a maximum of 8483 bits without error along with approximately PSNR 50.95 dB and improved the PSNR from 38 dB to 50.95 dB compared to the methods [23, 93]. In [105], Li et al., proposed block permutation along with stream cipher for encryption of cover media and bit replacement in prediction error for data embedding. Later on, Zhang et al. [21] proposed a separable RDH approach in the encrypted domain based on estimation. In this, $p\%$ pixels of the grayscale image were randomly selected by the content owner, which are denoted by I. The estimated value of these pixels was calculated from the existing $1-p\%$ pixels, which are denoted as E. But there was no pixel of E that surrounded the pixels set I. A threshold T was assumed to ensure the accuracy of pixel estimation. Only those pixels of I were chosen for estimation that had at least T pixels in E, which was denoted by I_a, and were not considered for estimation. on the other side, pixel of the set I surrounded by E, which had less than T pixel, denoted by I_b, was skipped and encrypted along with the $1-p\%$ pixels in E. After that, pixels in I_a were replaced by their estimating error. The estimating error can be either positive or negative, and it can be stored in an 8-bit pixel. The zero and one values of MSB indicate the positive and negative errors, respectively. To make the reversible of histogram, the error can be truncated between -124 and 125. If the error was

less than -124, then adjust it by adding 124, and if greater than 125, then it was subtract by 125. A location map was introduced to record these errors. Further, the embedding space was created by the modification histogram of errors. Qian et al. [106] proposed RDH in the encrypted domain based on a block cipher. In this, the cover image was divided into non-overlapping blocks of size 8×8. Each block was encrypted using AES with cipher block chaining mode. Further, secret information was embedded by modifying LSB of encrypted cover image. The probability of extracting the correct LSB was 0.5, and the PSNR was approximately 37.9 dB. In [113], Tang et al. introduced a differential compression approach for RDH to achieve high embedding capacity. Block based encryption approach was adopted to maintain spatial correlation among neighbouring pixels. Further, encrypted cover media was compressed by utilizing pixel correlation and creating free space for data hiding. Block of 3×3 was given best average compression ratio and average embedding capacity was 635532 bits. In [114], presented RDH based on intra block lossless compression approach for the encrypted domain. In this, the cover media was encrypted block-wise to preserve the correlation of the cover image, and then each block was permuted. Further, all blocks were divided into useful and non-useful blocks according to the correlation of blocks. Then, a non-useful block was reconstructed to create space for data hiding. By choosing an appropriate threshold, the average embedding capacity was 1,88,470 bits along with PSNR. This method outperformed the previous algorithms [93].

Li and Li [112] proposed a histogram shifting approach based on the homomorphic addition operation and secret data was embedded into the encrypted cover image after shifting the histogram. The embedding rate of this was approximately 1 bpp with low complexity. In [115], Jiang and Pang presented a Paillier homomorphic cryptosystem for encrypting a cover image. This has improved the efficiency of the encryption procedure. For data embedding, the difference expansion method was adopted, which exploits pixel pairs of

Table 2.6: Recent Advances on Reserving Room After Encryption

Author	Methodology	Database	Result	Remarks
Zhang [24]	Least significant bit compression	Self Generated	Embedding rate 0.017 bit per pixel(bpp) along with PSNR 39.0 dB	Embedding capacity increased with small block size
Xiao and Chen [107]	Compressive sensing	Self Generated	Embedding rate 0.05 bpp	Improved [24] embedding capacity & only one image used for experiment
Zhang et al. [101]	Pseudorandom sequence modulation	Self Generated	PSNR 37.9 dB	Payload capacity depends on the block size
Chen et al. [102]	Paillier public key cryptosystem , Homomorphic encryption	USC-SIPI	PSNR 42.85 dB, Payload capacity 0.25 bpp.	six images were used for the experiment for different payload
Wu et al. [97]	Joint and separable approach using Prediction error	Self Generated	Payload capacity 0.062 bpp and 0.156 bpp for joint and separable respectively.	separable approach outperforms Zhang's approach [24]
Zheng et. al [103]	Chaotic sequence, lossless compression, hamming distance	Self Generated	payload capacity 0.041 bpp, PSNR 55.0 dB	PSNR better compare to the methods [23, 24]
Yin et al. [104]	Multi-granular encryption, stream cipher, histogram modification	UCID	Payload capacity 8483 bits, PSNR 50.9 dB	This approach was suitable for tamper detection
Zhou et al. [108]	NTRU encryption	USC-SIPI	Data hiding capacity upto 0.8 bbp	Four image used for experiment
Qian et al. [109]	Slepian-Wolf source coding, Distributed source coding(DSC)	UCID	Embedding payload capacity approximately 0.2952 bpp	MSB plane was used to embed the secret message
Liu et al. [110]	Homomorphic encryption, Pixel prediction	BRODATZ, BOWS-2	Average payload capacity 0.74 bpp along with PSNR ∞ and SSIM one.	Improve embedding capacity and PSNR of method [111]
Li and Li [112]	Additive homomorphic, Histogram shifting	USC-SIPI	Embedding rate approximately 1 bpp & low complexity	Only color images used for experiment
Jiang and Pang	Additive homomorphic, Difference expansion	USC-SIPI	Payload capacity 1 bpp along with $+\infty$ and SSIM one.	Improved PSNR and Complexity of method [112]
Tang et al. [113]	Differential compression	BOWS-2, CVG-UGR	Average embedding rate 6,35,532-bits.	Block of 3×3 provided best compression ratio.

the cover image. Embedding rate was 1 bpp along with PSNR 57 dB of marked decrypted image, while PSNR of exact recovered image after data extraction tended to $+\infty$ with SSIM one. This method reduced the complexity but increased PSNR in comparison to Li and Li [112] scheme. Liu et al. [110] introduced RDH based on homomorphic encryption and prediction error for the encrypted domain. In this, the MSB value was predicted for each predictable pixel, and the MSB was used as redundant space. This redundant space was reserved after cover image encryption for data hiding. Further, the cover image was encrypted using a homomorphic multiplication encryption approach. Then, secret information was embedded into the encrypted cover image. The average embedding capacity of this approach was 0.74 bpp. along with PSNR ∞ and SSIM one. Recently, in [116] Ke presented a fully homomorphic encryption Encapsulated DE approach to encrypt cover media pixels and generate ciphertext. Secret data was embedded in the ciphertext by using the key switching LSB technique. Cipher-text expansion was controlled by the key switching method. The bootstrapping approach was used to control the noise during ciphertext decryption. Secret data was extracted from marked ciphertext or plaintext directly. In [98], Zhang et al. introduce RDH based on homomorphism to achieve high embedding capacity. In this, the cover image was divided into non-overlapping blocks and then encrypted using additive homomorphism. Further, the histogram shifting method was used to conceal data by using the prediction error. The average PSNR of the direct decrypted cover image was 41.95 dB, corresponding to an embedding capacity of 0.5 bpp, which was better than the existing method [19].

In 2020, [108], Zhou proposed a separable RDH in the homomorphic encrypted domain based on NTRU. In this, the cover image was divided into reference and neighbouring pixel groups by the content owner. Then, each group was encrypted by the public key. Next, the data hider divided the encrypted cover image in the same manner as the content

owner and determined the absolute difference for each group. Then, secret information was embedded into the encrypted cover image using the data hiding key. At the receiver end, secret information and cover image were extracted using the receiver data hiding key and private key, respectively. The embedding rate of this method was up to 0.8 bpp.

2.4.2 Reserving Room after Encryption using Specific Encryption

In this approach, cover media is encrypted using a specific encryption method so that after encryption, the correlation among the pixels is not removed completely. Li et al. [117] proposed the RDH method based on cross division and additive homomorphism. In this, the original cover image was divided into black and white disjoint sets. The cover image was encrypted using the encryption key. Further, a different histogram encrypted image was generated, and the histogram was shifted according to the secret information bit. Two adjacent boundaries were set to solve the overflow problem. Average PSNR was 34.41 dB along with embedding capacity of 0.5 bpp. Only eight images were used for the experiment. Further, Huang et al. [96] introduced a framework in which the cover image was divided into sub-blocks. Each sub-block of the cover image was encrypted using the same encryption key. These sub-blocks were randomly permuted. Twelve images were used for the experiment of size 512×512. The length of the location map was varied for different block sizes. Embedding capacity of this method was better than [23, 24]. In this sequence, Yi et al. [118] improved the embedding capacity of the Huang et al. [96] method. In this, the original cover image was encrypted with steps: block permutation and stream cipher. Since block permutation keeps strong spatial correlation and secret information bits were embedded into the image block using adaptive block-level based prediction-error expansion. Embedding capacity of this method was better than [96]. In [119] Zhang et al. proposed the RDH method based on reversible image transformation,

which improved the lossy method [120] in a reversible manner. In this, the cover image was transformed into another image of the same size. This transformation image was used as an encrypted image and data hider to embed data using any plain domain RDH method. Average Embedding capacity was 0.529 bpp, along with PSNR 27.2 dB, and a hundred images were used for the experiment from the BOSSBase database.

Further, [121] proposed RDHEI based on a redundant space transfer approach, in which redundant space was transferred from the original cover image to the encrypted cover image. To preserve the redundant space from the original cover image to the encrypted image Arnold transform encryption method was used. Then, secret data was embedded using the general RDH method. Embedding rate was approximately 1 bpp, while this rate was less than 1 bpp for other methods. In this sequence, in 2019, Qin et al. improved the [121] method by using redundancy transfer and a sparse block encoding approach [122]. In this, the original cover image was encrypted using a block-wise image encryption approach and redundancy transfer from $\lambda-$ MSB to $(8-\lambda)$-LSB simultaneously. Further, an improved sparse matrix encoding for different types of blocks within the LSB of the encrypted image was designed to create free space for secret data embedding. The average embedding rate was 1.72 bpp when the value of λ was taken as four.

Fu et al. [123] proposed the RDHEI method based on an adaptive encoding strategy. In this, the original cover image was divided into non-overlapping blocks, and these blocks were permuted. Then, the content owner encrypts each block, and the data hider embeds secret information bits into image blocks. PSNR values of the recovered image were infinity and better than schemes [24, 94]. In [124], Ge et al. proposed the RDHEI method based on block permutation, and the original cover image was divided into non-overlapping blocks. Then, blocks were pseudo-randomly permuted. Further, all blocks were encrypted, and the data hider selected peak pixels from each block. Secret data was embedded in each

block using histogram shifting. Embedding capacity was 0.15 bpp using the images from the UCID dataset. This approach obtained better embedding capacity in comparison to methods [23, 24, 96]. Later, S. Yi et al. [125] proposed a parametric binary tree labelling (PBTL) method to label pixels in two groups. Then, a data embedding algorithm was introduced based on PBTL, in which the spatial redundancy was used within the small blocks of the cover image for data hiding. The average embedding rate was 1.752 bpp, 2.003 bpp corresponding to block sizes of 2×2 and 3×3, respectively. This method was used to achieve redundancy only in small blocks of the cover image, but not in the entire cover. In [126], Liu and Pun proposed an RDHEI framework based on reversible image reconstruction(RIR). Then, RIR rearranges the original image to construct a redundancy image, and the meaningful information of the original image is used as an encrypted image.

Recently, Liu and Pun [127] proposed an RDHEI approach based on chunk encryption and redundancy matrix representation. In the chunk encryption approach original cover image was encrypted into chunks, and this retained the redundancy in the encrypted image. Redundancy matrix representation approach exploited the redundancies of the encrypted image to create free space for embedding the secret data. This method obtained better performance in comparison to methods [23, 105].

In Sum : In the recent past, some authors explored reversible data hiding based on specific encryption. In [117], Li et al. investigated RDHEI based on cross division and additive homomorphism, and two adjacent boundaries were set to solve the overflow problem. Further, Huang et al. [96] discussed an RDHEI method, which provided a better embedding capacity than [23, 24]. In this sequence, Yi et al. explored an adaptive block-level based prediction-error expansion approach to improve the embedding capacity of the Huang et al. method. In the same year, Zhang et al. investigated reversible image transformation and improved the lossy image transformation technique [120] in a reversible manner. C

.Qin et al. explored the redundancy transfer and sparse block encoding approach and improved the embedding capacity of the method [121]. Further, Fu et al.[123] investigated the RDHEI method based on an adaptive encoding strategy and improved the PSNR values of the schemes [24, 94]. Recently, Liu and Pun [127] introduced the RDHEI approach based on chunk encryption and redundancy matrix representation to enhance embedding capacity compared to [23, 105]. In the RRAE method, data extraction and media recovery are independent, and there can be few errors in the exercise of data extraction and/or image recovery. This issue was solved by reserving room before encryption, which is detailed in the following section.

2.4.3 Reserving Room before Encryption

In this, the content owner creates free space in the original cover media and then encrypts the cover media. After cover media encryption, free space is reserved for hiding the secret information by the data hider. This approach does an extra operation for reserving room before encryption at the content owner side. In 2013, Ma et al. [19] proposed the first RRBE approach. In this, the image is divided into two sets S and T, then LSBs of S were embedded into T with the existing RDH method [47, 128] such that the LSBs of S can be used for data embedding. Then, generate the encrypted image. The content owner divided the original image into a number of overlapping blocks, and then a function f was defined to measure the first order smoothness, and then selected the highest smoothness of blocks corresponding to the set S. Further, set T classified into white and black according to the pixel indices i and j satisfies the condition $(i+j)$ mode 2 =0 and $(i+j)$ mode 2 =1, respectively. Then, each white pixel was estimated by interpolation value using four neighbouring black pixels, and some data was embedded into the estimated error. In a similar way, estimating errors of black pixels using the neighbouring white pixels. These

two estimating errors are used for data embedding. After embedding the data, the watermark image was encrypted by a stream cipher. At the receiver end, after producing the marked decrypted image, the content owner extracts the data and recovers the original image. The process was done by the existing RDH methods [47]. In this scheme, the embedding rate with a single LSB-plane was less than 0.25 bpp, and more than 0.25 bpp for two LSB-planes. It is observed that the two LSB-planes give the significant embedding performance rather than the others. The proposed method was more than 10 times the payload capacity of the other methods [23, 24] in terms of PSNR. This method failed to select the best location for reserving space for data hiding. Mathew et al. [20] overcomes this limitation and achieves superior performance in terms of data hiding capacity. In this approach, different portions of the cover image are divided into small blocks to reserve space for secret information hiding. The blocks are categorized into active and smooth. Active blocks consisted of high pixel intensity, whereas smooth blocks had low pixel intensity. The LSB planes were extracted from the active blocks and embedded into the smooth blocks using the method [128], and the location of the LSB planes of active blocks was free for data hiding. Further, the image was encrypted using a stream cipher, and the locations of the LSB planes were already reserved for information hiding. Data hider encrypted the hidden data before hiding it in the encrypted cover image. This method gives the best result by the block size 32 and gains up to 3 dB PSNR at a higher embedding rate of the method [19]. In this sequence, Cao et al. [129] proposed patch patch-level RDH approach in the encrypted domain. In this, the cover image was divided into different non-overlapping patches, and only low residual error smooth patches were chosen for reserving free space. These patches were represented as the sparse coefficient, and the residual error corresponding to these patches was encoded and self embedded into non-selected patches. The cover image was encrypted via stream cipher such as RC4 or DES in cipher feedback

mode [130] to preserve the reserved free space and self embedding. In [131], Chen and Chang introduced RDH based on the run length coding method. In this, the first cover image was partitioned into blocks, and the MSB plane of each block was compressed by using run length coding to create free space. LSBs were embedded into this free space, and then the cover image was encrypted. The next secret information was concealed in the place of LSBs. Block size of 2×2 or 4×4 provided higher payload capacity and PSNR higher than 50 dB corresponding to payload capacity 1 bpp. Recently, in [132], Mohammadi et al. presented RDH in the encrypted domain based on pixel correlation on the block level. In this, the cover image was partitioned into non-overlapping blocks, and the centre pixel of each block was considered as the leader, while the others were followers. Prediction error was calculated between followers and the leader by using the difference predictions within blocks to analyze and tag blocks for hiding secret information. This tag was decided upon to hide data within blocks. At the receiver end, secret information was extracted along with the cover image by using the tag value. Embedding capacity was more than 1 bpp for block size 3×3, and this method outperforms the methods [125, 133].

In [134], Yin proposed an RDH method for encrypted domain based on multi MSB prediction and Huffman coding. In this, each pixel value was predicted using a median edge predictor with the help of three neighbouring pixels. Then, each bit of the original pixel and its predicted pixel was compared from MSB to LSB until a different bit was found, and, label of this pixel was determined as the number of same bits. next, the label map was constructed and converted into binary according to Huffman coding. Further, the cover image was encrypted using a stream cipher. Auxiliary information in the reference pixel was embedded into the encrypted image to ensure the Huffman coding rule and complete extraction of the label map. Data hider extracted auxiliary information from the encrypted image and embedded secret information accordingly. Embedding capacity was 2.583 bpp,

along with PSNR close to $+\infty$ with structural similarity one. This method performed better in embedding capacity than [133]. Malik et al. [135] proposed an MSB based RDH approach. In this, first, the original cover image was preprocessed by prediction error estimation techniques to create free space for data embedding. Further, the preprocessed image was encrypted using a stream cipher, and also, a location map was made to keep information about locations where secret data can be embedded or not. This location map was compressed losslessly using arithmetic coding and embedded in the LSB of the encrypted image. These original LSBs were MSB embedded into the MSB of the encrypted cover image. The encrypted image with location map information was sent to the data hider by the content owner. Secret information was embedded by replacing the MSBs assigned to the encrypted image. Average embedding capacity was 0.67 bpp along with PSNR 47.80 dB without location map and $+\infty$ with location map. In [136], Huang et al. introduced a specific encryption based RDH approach. In this, the prediction error was determined using a median edge detector to predict the pixel value, and then the cover image was encrypted by a specific encryption algorithm to create the free space. This encrypted image was permuted to enhance the security. After this, secret information was embedded into the encrypted cover using pixel value expansion. Embedding capacity of this approach was nearly 1 bpp along with PSNR ∞. UCID dataset images were used for the experiment. This method achieved a better embedding capacity than previous methods.

In [142], Puteaux and Puech presented a recursive huge-capacity RDH approach, which utilized spatial correlation between neighbouring pixels in cover images. In this, each bit plane from MSB to LSB was evaluated in order to focus on prediction errors, and the cover image was encrypted. If the prediction error was reasonable, then bits of the current bit-plane were replaced by secret information. The average embedding rate was 1.836 bpp and achieved 0.868 bpp gain compared to other methods [22] based on the MSB-plane.

Table 2.7: Recent Advances on Reserving Room Before Encryption

Author	Methodology	Database	Result	Remarks
Hong et al. [93]	improved [24]	USC-SIPI	error rate of lena image 0.34%, however in [24] was 1.21%, for block size	Improved [24] when block size was small, better estimation of smoothness of blocks
Yin et al. [137]	Separable, Error-Free	Self Generated	Payload for the block size 4x4, 5x5 and 8x8 were 0.129,0.126 and 0.114 respectively.	Maximum payload capacity much more than [23, 24] with best image quality
Qiu et al. [138]	Generalized integer transform, adaptive embedding	USC-SIPI	capacity 0.0009 bpp, PSNR 70dB	block size of 4×4 Embedding rate better than [139]
Puteaux et al. [133]	CPE-HCRDH, EPE-HCRDH	BOWS-2	CPE-HCRDH capacity 1bpp, PSNR = 46.87 dB, EPE-HCRDH capacity 0.9220 bpp, PSNR $+\infty$.	Number of error location in CPE-HCRDH and EPE-HCRDH 0.47 % and 0.46% respectively.
Zhou et al. [140]	Public key modulation mechanism, i.e, simple XOR operations.	Self Generated	21675 message bits embedded for block size 6×6 with 100% accuracy of data extraction.	Two class SVM classifier used to discriminate encrypted and non-encrypted image patches.
Qian et al. [99]	Improved [24], progressive recovery based upon separable RDH-EI	Self Generated	Average embedding rate approximately 0.0392 bpp, whereas embedding rate of [24] 0.0314 bpp	Three LSB layers used of the encrypted image. Average improvement of 23.7% of method [24] .
Zhang et al. [141]	Homomorphic public-key cryptosystems, Multilayer wet paper coding.	Self -Generated	The average embedding rate approximately $(1-1/2^K)$, where k denote the LSB plain for embedding.	Fifty grayscale images of size 1920 x 2560 used, Pailler cryptosystem used for pixel encryption
Ma et al. [19]	Reserving room before encryption	Self Generated	Embedding capacity more than 10 times of methods [23, 24] on the same PSNR.	Limited number of images used for experiment
Cao et al. [129]	patch-level sparse representation	Kodak, BOSS-Base, PASCAL, Holidays	Embedding rate 0.35 bpp, average PSNR for first three datasets were provided 5.2963 dB, 3.4312 dB, and 4.2360 dB	Embedding capacity approximately 1.7 times more than [19]
Zhang et al. [21]	Histogram shifting	Miscellaneous grayscale images and USC-SIPI	Embedding rate approximately 0.04 bpp and PSNR more than 55 dB.	This method does not reduced the sizes of location maps.

In this sequence, Puteaux and Puech [143] also introduced RDH based on a recursive method, which achieved an embedding rate was 2.458 bpp. This achieved better capacity than other existing MSB prediction based methods [22, 142], but it did not mention the security of secret information. Further, Dragoi and Coltuc [144] investigated the security of the Puteaux method and found the flows of unauthorised access to secret information and the content of the cover image.

Further, Wu et al. [145] improved RDH in the encrypted domain proposed by Yi et al's [125], in which spatial correlation was considered of the entire cover image rather than small blocks of cover to create free space for data hiding. Then, the cover image was encrypted by using the encryption key, and the encrypted pixels were categorized into two groups, in which one group was used for data embedding. Average embedding rates were 2.2683 bpp, 2.5613 bpp, 2.5194 bpp corresponding to datasets UCID, BOSSBase, and BOWS2, respectively, along with PSNR $+\infty$ and with SSIM value one. In [146], Yu et al. proposed an adaptive and separable RDH approach. In this, the cover image was partitioned into two segments that consisted of reference pixels and non-reference pixels. Next prediction errors of non-reference pixels were computed, and the original value of non-reference pixels was replaced by prediction errors. Further, the cover image was encrypted by using a stream cipher, and then secret information was embedded by modifying prediction errors. Embedding capacity was 0.433 bpp, along with PSNR 49.59 dB and with SSIM 0.9997. In [111], Xiang and Luo proposed an RDH scheme based on the homomorphic and Paillier cryptosystem. In this, the cover image was divided into groups of randomly selected adjacent pixels. Next, pixels of each group were categorized as reference and host pixels, and then the LSBs of reference pixels were reset. Further, the cover image was encrypted, and host pixels were replaced by reference pixels within the group to design mirroring cipher-text groups. Reference pixels in the mirroring cipher-text

group were used as reference, and encrypted secret information was embedded into the LSBs of host pixels by using homomorphic multiplication. At the receiver end, encrypted secret data was recovered by using the modular multiplication inverse. In [147], Wu et al. introduced RDH in the encrypted domain based on bit plane compression of prediction errors. In this, the first bit plane of prediction error was determined from the cover image, and then this bit plane was represented and compressed to create the free space for data hiding. Embedding capacity was better than proposed methods [133, 145], and three data sets were used for experiments. Table 2.7 summarizes the Performance results of the reserving room before encryption methods on different datasets.

In Sum: Reversible data hiding method based on reserving room before encryption has been widely explored to exploit spatial correlation to improve visual quality. In 2013, Ma et al. [19] investigated the first RRBE based on the LSB plane. This method was unsuccessful in selecting the best location for data hiding. Further, Mathew et al. [148] overcome this issue and improve the data hiding capacity. Later on, Cao et al. [129] introduced patch level RDH to preserve free space and self embedding. Chen and Chang introduced the LSB approach based on run length coding. Mohammadi et al. [132] explored correlation on the block level, and this outperforms [125, 133] methods. In [134, 135] investigated MSB prediction error-based methods to achieve a high embedding data rate. Puteaux and Puech [143] introduced a recursive data hiding method. But this does not mention the security of secret information. Dragoi and Coltuc [144] solved this problem. Yi et al. [125] introduced a parametric binary tree labeling based embedding method, which used spatial correlation for data hiding. But this does not fully utilize spatial correlation. In this sequence, Yu et al. [146] investigated an adaptive approach and improved capacity by dividing the cover image into blocks. In the next section, state-of-the-art dual image RDHs are discussed.

2.5 Dual Image Reversible Data Hiding

In this section, we discussed dual image reversible data hiding techniques. Various authors have been broadly investigating dual image RDH methods [149] to improve the data hiding capacity and security compared to the ordinary RDH. Dual image RDH provides a good solution when compared to the ordinary RDH methods. In dual image RDH, two copies of the original image are generated, then the sender hides the secret message in these two images using specific reversible data hiding methods, and two identical marked images are generated. These marked images are sent to the receiver end, and the receiver receives both. Then, the receiver extracts the secret message and recovers the original image from the marked images using a reversible data hiding algorithm. Attacker is unable to recover the secret message without obtaining both the marked image. Dual image RDH is also treated as a special case of secret sharing.

2.6 DI-RDH Classification Techniques

Dual image RDH methods have been significantly explored in the recent past to enhance data hiding capacity, security, and visual quality of the original cover image compared to the conventional RDH methods. Given this, we have categorized the dual image RDH task into exploiting modification direction, pixel value differencing, central folding strategy, and turtle shell. The details of dual image RDH work classification are outlined in Fig. 2.6.

Dual image RDH structure is instantiated in Fig. 2.7. In this, firstly, two copies of the original cover image are generated. Then, the secret message is embedded into both of these images, and two stego images are generated. Sender sends these two stego images to the recipient. The receiver receives both stego images at the same time and extracts

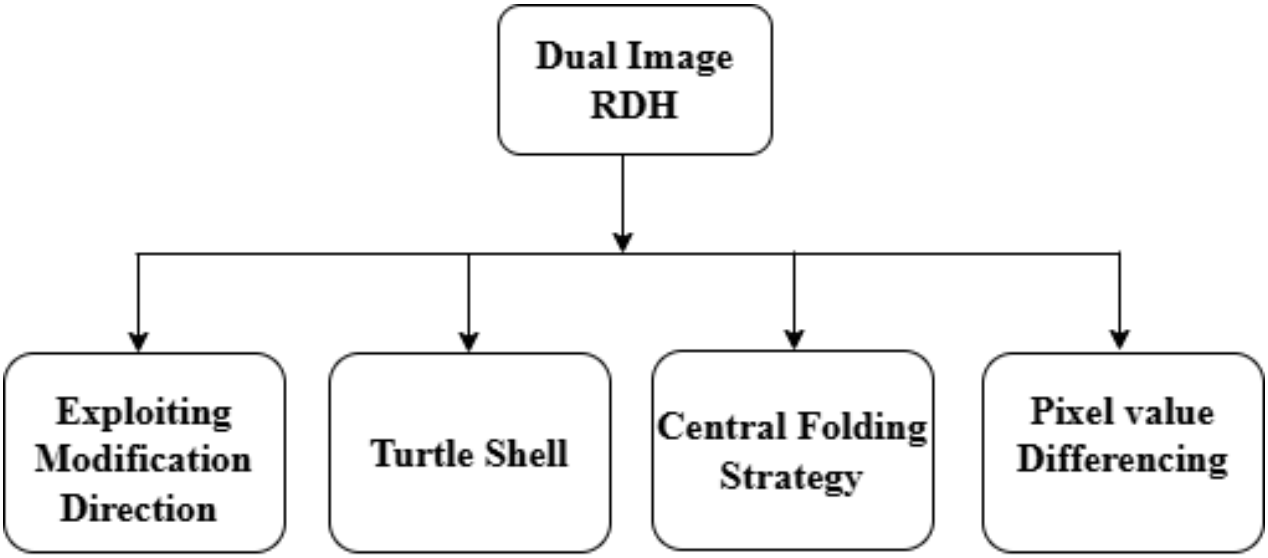

Fig. 2.6 Dual Image RDH Techniques

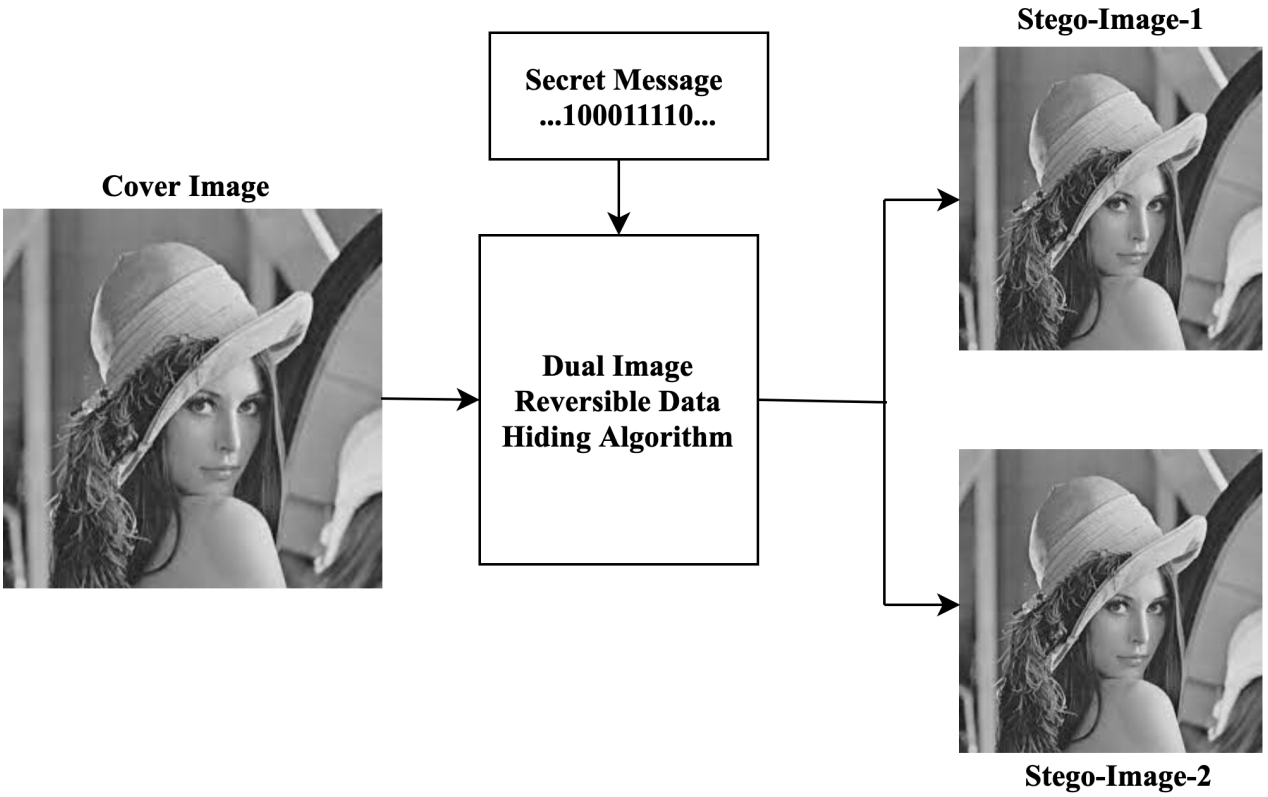

Fig. 2.7 General Framework of Message Embedding for Dual Image Reversible Data Hiding

the secret message along with the original image using the dual image RDH algorithm. Dual image work has been broadly investigated to obtain high data embedding capacity along with tunable image quality.

2.6.1 Exploiting Modification Direction

First Exploiting Modification Direction (EMD) [149] method for RDH was proposed in 2006. In this, firstly binary secret message is split into different chunks of L-bits, which are represented by k-digit in $(2n+1)$ base number system. Further, original image was partitioned into a sequence of n-sized pixel groups denoted as $(k_1, k_2, \ldots, k_n)$ and any perticular pixel is contributed in data hiding or not it decides by the function f such as $f(k_1, k_2, \ldots, k_n) = [\sum_{n=1}^{n}(k_i, i)]$ mod$(2n+1)$. Every message bit d can be hide into set of n pixels $(k_1, k_2, ..., k_n)$ by using the three embedding steps:

1. If $d = f$, then embed d without modifying pixel group.

2. if $d \neq f$, determine $s = d - f$ mod $(2n+1)$. If $s \leq n$, then embed d, and the value of k_s is incremented by 1.

3. If $d \neq f$, determine $s = d - f$ mod $(2n+1)$. If $s \leq n$, then embed d, and the value of k_{2n+1-s} is incremented by one.

Further, the EMD approach was introduced for dual-image RDH in order to achieve high security of secret information and better visual quality of cover image by Chang et al. [150] in 2007. In this, two secret digits in quinary form were embedded into the original pixels and encoded into two steganographic images. The average PSNR of both stego images was higher than 45 dB, and data concealing capacity was 1 bpp. This method outperforms the method proposed in [149]. In a similar line, Chang et al. [151] also increased the visual quality of stego images by converting the 5 x 5 matrix into horizontal and vertical directions rather than diagonal. Except for the directions of coordinates, method [150] and [151] were similar. This method kept the visual quality (i.e., PSNR) of both stego images above 48 dB on average, and the embedding capacity was 1 bpp. Further, Lee and Huang [152]

proposed RDH based on a combination of pixel orientation located in two stego-images to improve data hiding capacity. In this, firstly secret binary sequence was converted into a sequence of digits with base five. then, two consecutive digits d_1 and d_2 were embedded into the cover pixel pair for each stego-image, and the pair of pixel values was mapped in two dimensions. However, this method can embed only five secret bits in each set of four cover pixels, so it restricts the embedding data. The average embedding capacity and PSNR of the two were 1.07 bpp and 49.73 dB, respectively. This method provided better data concealing capacity and PSNR compared to the method [151]. In this order, Chang et. al [151] overcome the data embedding restriction for the Lee and Huang method [152] by using the magic matrix technique to improve the data hiding capacity. Magic matrix was generated using formula $F(s_1, s_2) = (s_1 + 3 \times s_2)$ mod 9 [**?**], where, s_1, s_2 represents pixels pair and function $F(s_1, s_2)$ denote the message digit. If the function $F(s_1, s_2)$ value is not equal to the embedded secret data. Then, the current pixel pair was changed to its two upper left or two bottom right pixel pairs to find the value, which is equal to the embedded secret data. Embedding rate and PSNR were 1.55 bpp and 39.89 dB, respectively, for both the stego images. In [153], three rules were given for data concealing into the first stego-image, while data embedding in the second stego-image depends on rules generated in the first stego-image. Further, Lin et al. [154] introduced five base EMD method, in which two secret digits are embedded together in every pair of pixels of the original cover using the EMD matrix. Recently, Chen et al. [155] developed an EMD method for dual image RDH using a reference matrix, and each pixel was carried $1 + \log_2 5$ bits along with horizontal and vertical directions. Embedding capacity was achieved upto 1.56 bpp and PSNR below 42 dB, and provided the better embedding capacity in comparison to [152, 156] methods.

In Sum: Dual-image RDH based on Exploiting Modification Direction is extensively ex-

plored. Initially. Chang et al. [150] introduced the EMD approach based on quinary digits. In this sequence, Chang et al. [151] presented 5×5 based method by using horizontal and vertical directions in order to increase the visual quality of [150] from PSNR 45 dB to 48 dB. Further, Lee and Huang [152] investigated DI-RDH based on a combination of pixel orientation and improved slightly payload capacity and visual quality compared to proposed methods [150, 151]. Later, Chang et. al [151] Recently Chen et al. [155] introduced reference matrix based DI-RDH and provided a better embedding capacity in comparison to [152, 156] methods. In the next section turtle shell method for DI-RDH is discussed.

2.6.2 Turtle Shell

Turtle shell (TS) consisted of eight distinct values from zero to eight, in which six edge digits and two backwards digits are included. Chang et al. [157] introduced the first turtle shell based DI-RDH approach in 2014. In this, the secret bitstream was partitioned into 3-bit chunks, and the value of each chunk was converted into secret digits. Then, the reference matrix M was created to embed the secret digits. For this, the difference of two adjacent pixels was calculated and it was denoted by the number 1 for the same row, whereas 2 and 3 for alternate columns. In this way, the reference matrix M was generated by collecting the number of neighbouring hexagons, and each hexagon of the reference matrix is called TS. Further, associated set G for matrix $M(x_i,x_{i+1})$ was generated according to the following rules: a) If $M(x_i,x_{i+1})$ was back wards digit within turtle shell, then corresponding set G was consisted all digits in same TS. b) If $M(x_i,x_{i+1})$ was an edge digit presented in at least one TS, then the corresponding set G consisted of all digits of the turtle shells in which $M(x_i,x_{i+1})$ belongs. c) If $M(x_i,x_{i+1})$ was not presented in any turtle shell, then the corresponding set G consisted of all digits in a 3×3 sub-block where $M(x_i,x_{i+1})$ was located. Pixel pair (x_i,x_{i+1}), was selected from the reference matrix $M(x_i,x_{i+1})$ to hide

the secret digit d_j, if its distance was minimum among all pixels of reference matrix and modified from (x_i, x_{i+1}) to (x'_i, x'_{i+1}) selected from all the elements of $M(x_i, x_{i+1})$. At the end of the receiver end, the receiver extracted the secret message without any error from the stego image. For this„ every stego pixel pair (x_i, x_{i+1}) was mapped to $M(x'_i, x_{i+1})$, then value of $M(x'_i, x_{i+1})$ corresponding to reference matrix M was determine as secret digit. Eight grayscale images were used for the experiment. The average EC and PSNR of this approach were 1.5 bpp and 49.40 dB, respectively, and higher than the proposed method [149]. In [158], Xie et al. presented a DI-RDH approach based on a two-layer turtle shell matrix, in which a 4-ary digit is assigned to each member of the TS matrix corresponding to symmetrical distribution. Two extra bits were concealed into each pixel pair compared to the original TS method [157]. Further, [156] discussed the DI-RDH approach that applied the turtle shell reference matrix to conceal a message in two shadows. In this, a satisfactory quality of the generated shadows was obtained. The data hiding capacity was approximately 1 bpp. In this sequence, a real time DI-RDH approach using a new TS reference matrix to embed secret information was presented in [159]. For this, two shadows were used for data embedding, in which the first one was used for one bit embedding while the second was used for four secret bits embedding. Embedding capacity was higher than 1.25 bpp. It also resisted static attacks of pixel value differencing histogram and was used in numerous real time. applications. In a similar line, Xie et al. [160] also proposed a novel turtle shell dual image RDH approach based on a reference matrix to achieve a high embedding capacity. This divides a pixel pair duplicated from a pixel into the upper category and the lower category. In accordance with the pixel pair category, a precise pixel block of size 4×5 was selected; this pixel pair was used so that each pixel pair holds a 16-ary secret digit. This method provided a better embedding rate compared to the state-of-the-art methods [152, 156, 159]. PSNR was inferior for these methods.

In Sum: Recently, researchers investigated the Turtle shell based DI-RDH method. Chang et al. [157] explored the Turtle shell method based on the reference matrix and increased the payload capacity as well as visual quality in comparison to [149]. Further, Xie et al. [158] improved the embedding capacity of the Chang et al. [157] method by using two layer turtle shell matrix. In this sequence, Liu and Chang [159] introduced a real time DI-RDH based on the TS reference matrix, which resists static attacks of pixel value differencing histogram. In a similar line, Xie et al. [160] improved the payload capacity of the presented state-of-the-art methods [152, 156, 159] with inferior PSNR by using the turtle shell approach. In the next section central folding strategy method for DI-RDH is discussed.

2.6.3 Central Folding Strategy

Central Folding Strategy (CFD) technique compresses the secret information into digits using a different numeral system. Then, the average method is applied to conceal the compressed message in two stego-images. Lu et al. [161] introduced the first CFD approach in 2015. In this, k-message bits were converted into digits d using the numeral system. Then, digit d was changed into folded secret information $\bar{d}$ using $\bar{d} = d - 2^{k-1}$. Further, digit $\bar{d}$ divided into digits $\bar{b_1} = \lfloor \bar{d}/2 \rfloor$ and $\bar{b_2} = \lceil \bar{d}/2 \rceil$ and these were used for data embedding as follows: $t'_{i,j} = t_{i,j} + \bar{b_1}$ and $t''_{i,j} = t_{i,j} + \bar{b_2}$. At the time of message extraction, if stego pixels $t'_{i,j} = t''_{i,j}$ and $t'_{i,j}, t''_{i,j} \in [2^{k-1}, 256 - 2^{k-1}]$, then these pixels were not contained any secret information. If stego pixel $t'_{i,j}$ or $t''_{i,j} \in [2^{k-1}, 256 - 2^{k-1}]$, then these pixels were contained secret information and it can be extracted as follows: $\bar{d} = t'_{i,j} - t''_{i,j}$ and $d = \bar{d} + 2^{k-1}$. This method does not use the correlation between the adjacent digits to minimize the distortion. Average embedding capacity for $k = 2, 3$ were 524204 and 785204, respectively. This method provided better data hiding capacity and PSNR compared to the proposed methods [150, 151, 152]. In this order, H. Yao [162] improved the Lu et al. [161] method

by adopting the shift strategy technique. For this, the bitstream of the message was partitioned into different sets of k bits chunks and then converted these chunks into decimal digits. If digit $d = 2^k - 1$ then, digits d was updated using $d = d + m_{k+1}$ by adding an extra m_{k+1}^{th} bit. Secret digit was embedded into the cover pixel and stego images were generated as follows: (1)if d is an even integer, then $x_i' = x_i + \lfloor d/4 \rfloor$ and $x_i'' = x_i - d/2$. (2) if d is odd integer, then, $x_i' = x_i - \lceil d/4 \rceil$ and $x_i'' = x_i' + \lceil d/2 \rceil$. One bit was extra embedded whenever the condition $d = 2^k - 1$ occurs compared to the method Lu et al. [161].

Further, Chi et al. [163] proposed the CFS method based on a dynamic encoding strategy. In this, a set of k secret information bits set $b_1, b_2, , ..., b_k$ was converted into decimal values d_i. then, a code book of size 2^k was prepared to encode the decimal values d_i . Codewords and their indices were sequentially stored from 0 to $2^k - 1$, and the indices of codewords were replaced by digit values of secret information. This decreases the distortion of the stego images. Average embedding capacities were $1, 1.5$ and 2 for $k = 2, 3, 4$ along with PSNR 49.14 dB, 49.23 dB, and 49.40 dB for stego images, respectively.

In Sum: the Dual-image reversible data hiding method based on the centre folding strategy has been widely explored to increase the embedding capacity and improve the visual quality of the cover image. In 2015, Lu et al. [161] introduced the first central folding strategy using the d-base number system. This method does not use the correlation between adjacent digits to minimize the distortion. In this order, H. Yao [162] introduced the Lu et al. [161] method by adopting the shift strategy technique. Further, Chi et al. [163] investigated a dynamic encoding strategy to improve the payload capacity of [161, 162]. In the next section pixel value differencing method for DI-RDH is discussed.

2.6.4 Pixel Value Differencing

Pixel Value Differencing (PVD) was first introduced by Wu and Tsai to differentiate between smooth and edge regions, but this method was not reversible. In 2017, Biswapati et al.'s [164] suggested a first DI-RDH strategy based on PVD and EMD. In this, an original image of size $H \times W$ was interpolated and generated a new image of size $2H \times 2W$. Secret information was partitioned into a stream of 7 bits, and each stream was again segregated into two sets, where 4 bits and 3 bits were concealed using PVD and improved EMD, respectively. Then, the modified two pixels were obtained, containing seven secret information bits, and these pixels were distributed between the dual images using a secret key. The receiver extracted the secret information and recovered the original image with the help of the secret key. This scheme provided 1.75 bpp data hiding capacity and PSNR of more than 37 dB. Later, in 2019, Jung et al.'s [165] introduced a DI-RDH approach using the PVD, and the cover image was divided into non-overlapping sub-blocks. The highest difference value in each sub-block was calculated. To determine the embedding size, the highest difference and the log function were used. Then, the secret message was converted from binary bits to decimal secret data. Decimal message digits were concealed in the cover image by using the floor function and the ceil function to generate two stego images. At the receiver end, the cover image was recovered by averaging the pixel values of the two stego-images. The secret data was extracted by calculating the maximum difference value and the log function within each block of the cover image. This scheme has an inferior PSNR with the methods [166, 167] with low payload; however, when the payload was increased, PSNR was higher than other schemes. The average PSNR of the proposed scheme was 42.28 dB and 50.96 dB when the payloads were 80,000 bits and 500,000 bits, respectively. In this sequence, LSB matching and modified PVD with n-Rightmost Bit Re-

placement (n-RBR) approaches were proposed in 2019 by A.K. Sahu and G. Swain [168]. In LSB matching, two identical images were used for data hiding, whereas the modified PVD approach used four identical images for data hiding. LSB matching improved the PSNR of [169] at the same level of embedding capacity. The average data hiding capacity of this approach was 524, 288 along with 51.18 and 51.17 dB PSNR. In modified PVD, data embedding is performed in two phases, and four identical images of the original image are used for data hiding. In phase 1, firstly, two pixels from each identical image $I_1(P_1, P_2)$ and $I_2(Q_1, Q_2)$ and two different sets x_1 and x_2 of n-bits secret data were taken. Then, n-LSBs from the pixels P_1 and Q_2 and binary sets x_1 and x_2 were converted into the decimal numbers, namely p_1, q_2, b_1, b_2 respectively. Further, difference value these four values were calculated i.e $d_1 = p_1 - b_1$ and $d_2 = q_2 - b_2$ and three modified pixels P_1', P_1'', P_1''',were determined corresponding to pixel P_1 such that $P_1' = P_1 - d_1, P_1'' = P - 1' - 2^x, P_1''' = P - 1' + 2^x$. Then, compute difference values $d_3 = |P_1 - P_1'|, d_4 = |P_1 - P_1''|, d_5 = |P_1 - P_1'''|$ and with help of these difference values stego pixel P_1^* for P_1 were generated. Similarly, stego pixel Q_2^* for Q_2 was also generated. In phase 2, modified PVD was used for embedding in the last two identical images. for this, two pixel from image $I_3(r_1, r_2)$ and two from $I_4(s_1, s_2)$ were chosen. Then, the difference d_6 was determined between pixel r_2 and S_1, and the number of embedding bits x can be calculated using this difference value by determining the value of $\lfloor \log_2 d_6 \rfloor$. Further, two sets x_1 and x_2 with x of secret information bits were converted into decimal numbers b_3 and b_4 and values $b_3' = \lfloor b_3/2$, $b_3'' = \lceil b_3/2 \rceil$, $b_4' = \lfloor b_4/2$, $b_4'' = \lceil b_4/2 \rceil$ were determined. At last, stego pixels were calculated using $r_{1*} = s_1 + b_3'$, $r_{2*} = r_2 + b_4'$, $s_{1*} = s_1 - b_3''$, $s_{2*} = r_2 - b_4''$, and the stego marked image was generated. The average data hiding capacity of this approach was 796,729 along with 54.16, 54.16, 38.32, and 37.50 dB PSNR for $n = 1$.

In Sum: Dual-image reversible data hiding method based on pixel value differencing has

been explored recently. In 2017, Biswapati et al.'s [164] investigated the first PVD and EMD based approach. Further, Jung et al.'s [165] investigated and outperformed compared to Biswapati et al.'s [164] method. In 2019, A.K. Sahu and G. Swain introduced LSB matching and modified PVD, which improved state-of-the-art methods [164, 165, 169]. In the next section analysis of DI-RDH methods is discussed. Analysis of Dual-image RDH Methods Dual image reversible data hiding is mainly classified into Exploiting modification direction, turtle shell, Central folding strategy, and Pixel value differencing methods. Each method of DI-RDH has its advantages and disadvantages, and the superiority of a method depends on the specific requirements and constraints of the applications. Exploiting modification direction is used for high embedding requirements, whereas center folding and turtle shell methods can be more suitable to obtain high quality images along with reasonable payload capacity. Priority pixel value differencing method may be the best option to achieve low distortion and security with high payload capacity, but the complexity of this is more compared to the other methods. In general, DI-RDH methods can be used according to specific requirements and constraints of the application. In the next section, benchmark datasets for RDH are discussed.

2.7 Benchmark Dataset for Reversible Data Hiding Technique Evaluation

In the last two decades, several algorithms have been proposed by various authors in different journals. Most of the reported work has been evaluated on a different dataset. Hence, to provide a common base for the evaluation of various reversible data hiding algorithms and to give readers a quick overview of various publicly available datasets, we have briefly summarized various datasets. For the sake of clarity, we have tabulated the

various datasets in Table 2.8, and their brief description is given as follows:

- USC-SIPI database [49] contains textured 154 monochrome images in which 129 are of size 512×512 and 25 of the size 1024×1024. This includes 38 aerial images, of which 37 are colored with 12 of size 512×512, 24 of size 1024×1024, and one monochrome. It consists of 40 miscellaneous images and a sequence of 69 monochrome in which 3 sequences of 16, 32, and 11 images of size 256×256 and one sequence of 10 images of size 256×256.

- Initially BOSSBase database [62] was published in three stages. The first version, 0.90, was released in June 2010. This version contained 7518 images. The second version, 0.92, was published with 9074 images. Finally, version 1.0 was released in May 2011 and contained 10000 images. All images were captured from 7 different cameras. This database consists of color images with full resolution in RAW format. All the images were resized into 512×512 pixels, and then converted into a grayscale image with size 512×512.

- The BossRank database consists of 1000 grayscale images with size 512×512, in which 847 images were captured by Leica M9 in RAW form and 153 images from Panasonic Lumix DMC-FZ50 in JPEG format.

- Kodak dataset [61] consists of 25 uncompressed PNG color images. The size of each image 768×512 pixels.

- The holiday dataset [172] consists of 500 image groups, and every group represents a different object. Each group has one query image as its first image. This data set produced 4455091 descriptors with dimension 128 and also includes 1491 images.

Table 2.8: Benchmark for Evaluation of Reversible Data Hiding Techniques

Data Set	Used For Dataset	Free/Paid	Details	Remarks
USC-SIPI [49]	[8, 21, 40, 48, 58, 60, 68, 72, 76, 81, 89, 93, 95, 102, 108, 112, 114, 115, 116, 132, 138]	Free	Each black and white image pixel consists 8 bits and color image pixel with 24 bits	Images of this dataset are available in TIFF format
BOSSBase [62]	[35, 40, 58, 73, 81, 125, 129, 131, 134, 145, 147, 170, 171]	Free	Three versions published and contained a total of 26592 images.	Images captured by different cameras and converted into grayscale images.
Kodak [61]	[58, 73, 116, 125, 129]	Free	25 uncompressed PNG color images	Lossless true color image suite
Holiday [172]	[129]	Free	500 images groups, total 1491 images	This includes high resolution images with different varieties such as natural, water and fire effects, etc.
CVG-UGR [173]	[35, 75, 113, 135]	Free	Include 235 grayscale images of different sizes, containing astronomical and color images for different sizes.	Mostly images in the Portable Bitmap Format (PBF) and Portable Graymap Format (PGF).
UCID [174]	[71, 104, 109, 134, 136, 145, 146, 147]	Free	1338 uncompressed color TIFF images	Include natural picture, man-made objects.
Brodatz [175]	[75, 110, 134]	Free	Total 32 texture classes, every class contains 64 images with size 64×64	All images stored in khoras visualization(xv) format
Corel Draw [175]	[9]	Paid	68,040 images, four features extracted from each image.	Feature set of any image stored in different files

- Corel Draw dataset [176] consists of 68,040 total images in various categories. Each feature set was stored in a different file and consisted of a line corresponding to an image for each file. The first value of the line denotes ID, and the other subsequent values were the feature vector of the image. Four features, such as color histogram, color histogram layout, color moments, and co-occurrence texture, were extracted from each image.

- CVG-UGR [173] is an image database which contains grayscale images that is illusory of size 128×128 total 25 images, miscellaneous of various sizes are 6 images, miscellaneous of size 256×256 are 68 images and 512×512 are 96 images and bigger than size 512×512 are 4 images and contours of different sizes are 36 images. It also contains the Biomedical, nematodes, and nematodes 3D slides, astronomical and color images of different sizes. Most of the images are in Portable Bitmap Format (PBM) or Portable Graymap Format(PGF).

- UCID [174] is an uncompressed color image data set, which contains 1338 uncompressed TIFF images. This includes natural pictures and man-made objects. These images were taken by a Minolta Dimage 5 digital color camera. It is an alternative dataset for the evaluation of the image extraction approaches, and also provides grading of image compression and color quantisation methods.

- The normalized Broadatz Texture database contained 2048 images, which are divided into 32 texture classes. Every class includes 64 images, of which 16 are original, 16 rotated, 16 scaled versions, and 16 rotated and scaled versions of the original images with size 64×64 pixels. The format of all the images is Khoras visualization(kv).

2.8 Conclusion

Reversible data hiding has been extensively investigated mainly due to its potential applications in the diverse digital world. In this chapter, we have performed an exhaustive review of recent developments in the field of reversible data hiding. For this, reversible data hiding methods are precisely categorized mainly into the plain domain and encrypted domain. Further research work is subcategorized, wherein the merits and demerits of each class are highlighted. For instance, RDH work in the field of plain domain is further categorized as a) Lossless Compression, b) Expansion, c) Histogram Shifting, d) Interpolation Techniques. On the other hand, encrypted domain work is classified as: a) Reserving Room after Encryption b) Reserving Room before Encryption. In addition, a tabular comparison of recent developments brings out a research gap in the field of reversible data hiding. Performance of state-of-the-art reversible data hiding techniques is analysed via performance metrics, namely embedding capacity, imperceptibility, and computational complexity. Generally, a trade-off exists between the quality of cover media and the capacity of data. Lossless compression-based reversible data hiding methods have limited embedding capacity mainly due to overhead data embedding along with secret data. Minimization of overhead data is a challenge due to its requirement for faithful recovery at the receiving end. Although reversible data hiding methods based on histogram shifting can resist several security attacks, their application in actual deployment scenarios needs a pragmatic approach to address the issues. Separable reversible data hiding methods can only recover the secret data and the cover media. However, the independent recovery of cover media and secret data without any error is still partially solved. We have also reviewed the state-of-the-art dual image reversible data hiding methods. These are categorized as exploiting modification direction, pixel value differencing, central folding strategy, and turtle shell, and

also explore their merits and demerits. Performance of various dual image reversible data hiding methods is compared in the context of data concealing capacity and visual quality of stego images.

Chapter 3

Design and Development of Plain Domain RDH Techniques

3.1 Introduction

RDH is broadly used in defense, stereo image coding, medical, law enforcement, image authentication, etc., since distortion in the original cover is not tolerable for these scenarios. Up to now, many RDH techniques have been explored by various researchers in different domains. RDH methods mainly focus on obtaining a high data hiding capacity as well as faithful recovery of the cover and securely transferring the secret information over the communication channel. In the plain domain, the data hiding process is carried out directly on the original, unencrypted host medium. However, most of the techniques suffer due to concerns about low data capacity, data security, and mainly in handling the communication overhead. The communication overhead was transferred through a separate channel. To overcome these issues, a multistage high-capacity reversible data hiding technique without overhead has been developed in this chapter, where histogram peaks are exploited for embedding the secret data in the plain domain and further making it an

encrypted marked cover media. Our main concern, i.e., the overhead bits are embedded in the encrypted marked image hereby, using both the plain domain and encrypted domain. Thus, the aim of not sending any auxiliary information to the receiver directly and enhancing the embedding capacity has been fulfilled. To further detail, the peak of the histogram of the cover is first determined. Then, the histogram is shifted according to the embedding bits. Histogram shifting creates an overflow problem, and to control this problem, we record pixels with their indices, known as the communication overhead information. Further, secret information is embedded into the plain domain, which generates the marked cover. This marked cover is further encrypted using a specific algorithm, which maintains the correlation among the encrypted pixels. Peaks of the encrypted marked image are used to hide the communication overhead information of both the plain and encrypted domains. This communication overhead information helps the receiver to extract the secret information and to ensure the recovery of the cover image. In the proposed technique, we embed two types of data in different domains. Secret information is embedded into the plain domain, whereas communication overhead information of both is embedded in the encrypted domain. Information embedded in the encrypted domain helps the receiver to extract the secret information from the plain domain. However, most of the plain domain RDH techniques suffered due to low embedding capacity, though the overhead information and security of the secure message were rectified. To overcome this problem, a bidirectional multistage embedding technique has also been proposed in this chapter wherein the plain cover image is used to embed the secure data, and an encrypted cover is used for embedding the overhead information, which improves the data hiding capacity as well as the security of the secure data. In addition, no separate channel is required for communication overhead transmission. Bidirectional secret data embedding is performed through the exploitation of both left peak and right peak from the main histogram peak. For this,

all pixels after the right peak are shifted $(2^n - 1)$ - place right, and accordingly, all pixels before the left peak are shifted $(2^n - 1)$ -place left, wherein n denotes the number of secret data embedding bits per pixel. Experiments are performed on various sets of images, and results reveal better performance over the state-of-the-art methods. In this chapter, multistage and bidirectional reversible data hiding techniques are discussed in detail, along with their experimental results.

3.2 Related Work and Its Research Gap

Reversible data hiding has been extensively explored recently due to its numerous applications in different fields. Generally, RDH work can be categorized into lossless compression [2, 3, 114], difference expansion [6, 44, 45], prediction error expansion [58, 134, 147], histogram shifting [65, 177, 178, 179, 180, 181, 182], and interpolation [82, 84]. These works are discussed in detail in the review paper [183].

Histogram shifting is a promising area which have been investigated in different directions. It is closely related to the proposed work. Recent evolution in the domain of histogram shifting for RDH is summarized as follows. Firstly, Ni et al.[177] proposed the RDH method based on histogram shifting in 2006. In this, the maximum frequency of pixel, i.e, peak value p, and pixel with the lowest frequency, i.e, zero point z of histogram were determined. Then, the image traversed pixel by pixel, and if the pixel was in range of $[p+1,z]$, then it was shifted one place to the right. If pixel was found to be equal to p, then the current data bit was added to it. The capacity of this method is equivalent to its peak values. The pixels that were overwritten during shifting were transmitted with a location map through a separate channel. At the receiver end, when a pixel in the received image was found to be equal to p, then the extracted bit was 0, and when found $p+1$ then it was 1

extracted. Fallahpour and Sedaaghi [65] further improved the Ni et al. [177] method by introducing a block-wise histogram shifting. In this, the cover image was divided into blocks, and each block was treated as a separate entity for histogram shifting enabling more data hiding. Then, for a given block of n pairs, the peak point p and the zero point z were calculated. In i^{th} pair of (p_i, z_i) if $(p_i > z_i)$, then pixels in range $[z_i + 1, p_i]$ were shifted 1 place to the left. Further, data hiding was performed according to the method [65]. For data recovery, the image was traversed, and if values $p_i - 1$ and p_i were encountered, then data bits 1 and 0 were recovered respectively. If $(z_i > p_i)$ then the pixels in range $[p_i + 1, z_i - 1]$ were shifted one place to the right and the image can be recovered according to methods [65, 177]. Ying et al. [179] proposed histogram RDH to improve the contrast and overall visual quality of the image. In general, histogram shifting minimally alters the pixel values, which helps to maintain the structural and visual integrity of the image. Qin et al. [180] presented a two dimensional histogram approach that divided the histogram of the cover image into two separate histograms. This approach efficiently increased the contrast by utilising the bi-histogram equalization. Reversibility was achieved through inverse mapping and changes during the histogram equalization process were recorded. Similarly, Q. Chang et al. [181] proposed an adaptive method for color image through three dimensional histogram modification. In this, the correlation among the Red, Green, and Blue color channels was determined. This correlation was utilised to hide the secret data, and the adaptive nature of the method ensured that the visual quality of the stego image remained high, along with minimal distortion. This method was specially designed for those applications that deal with colour images, and preserving color fidelity was crucial.

Further, Kumar et al.[182] improved Ni et al.[177] and Fallahpour and Sedaaghi [65] methods. For this, a multistage RDH method was proposed to enhance the data hiding capacity. In first stage, secret message was hidden in the same manner as mentation

in [65, 177], but the communication overhead was embedded in the second stage i.e, encrypted image. Image was encrypted using an affine cipher to maintain the same or higher maximum peak of histogram. This method addressed the challenge of separately communicating overhead data by introducing flag-based marking within the cover image itself after data embedding. Further, peak value p and its overhead information along with encrypted peak and overhead information were embedded in the peaks of encrypted image. Receiver distinguish the cover and encrypted image overhead information by the identification of the flags. Further, M. Xiao et al. [178] proposed RDH using multiple peak and zero bins to embed secret data. For this, pixel values in the peak bins were shifted to adjacent zero bins creating free space for secret data hiding. This approach increased the capacity for hidden data while maintaining high visual quality of the marked image.

In sum, histogram shifting was explored extensively to achieve high embedding capacity without degradation of the quality of the image. Still, achieving the high capacity along with security is a challenge. Also, communication overhead needs a separate channel, which may lead to a compromise in security. To address this, we proposed a Multistage Embedding without communication overhead and Bi-directional Histogram Shifting with Multistage Embedding RDH methods. The core design of the proposed method is presented in the subsequent sections.

3.3 Core Design of Multistage without Communication Overhead RDH

The proposed method multi stage reversible data hiding method, is an effective technique designed to enhance data hiding capacity, which consists of multistage embedding. The aim of the proposed method is to communicate the secret message without sharing the

auxiliary/overhead information to the receiver through a separate channel. The general framework of the Proposed approach is shown in Fig. 3.1. Fig. 3.2 represents the flowchart for data embedding, whereas Fig. 3.3 represents the flowchart of secret data extraction and cover recovery. Details of the proposed data embedding, extraction, and recovery of the cover image are discussed in sections 3.3.1 and 3.3.2, respectively.

3.3.1 Data Embedding

Data embedding is performed by utilizing the peak of both the plain and encrypted images. In this method, the cover is converted into pixels and then a histogram of the cover is generated. Then, the cover image is scanned to determine the maximum point P_1 i.e, pixel value with maximum occurrence, and the minimum point Z_1 means such pixels do not exist in the image or have the minimum number of occurrences. Then, row and column indices values of pixels lying between $[Z_1, Z_1 + (2^n - 1)]$ are stored with pixel value as an overhead O_1 of the original cover image or increased values with $2^n - 1$ places until it becomes greater than 255. Further, pixels that lie between $\{P_1, Z_1\}$ are shifted upto $2^n - 1$ places. After that, binary sequence say $b_1, b_2, b_3, ..., b_n$ of the secret message is divided into number of chunks of n-bits say $n_1, n_2, n_3, ..., n_k$, and these chunks n_i 1 to k are separately converted into decimal number sequence $d_1, d_2, d_3, ...d_k$. Then, the original cover image is again scanned and whenever peak P_1 is encountered, the peak value P_i is incremented by d_i number and the marked cover image is generated. Further, the marked cover image is encrypted using a specific cipher so that the correlation of pixels is also maintained in the encrypted image. Encrypted image is scanned to determine the maximum point P_2 and zero-point Z_2, and the shifted pixels lie between $\{P_2, Z_2\}$ with one place. Then, the peak of the encrypted image is used to embed overhead data O_1 of the original image and O_2 of the encrypted image. After embedding overhead data O_1 in the encrypted image, if the

peak of the encrypted image has a sufficient number of counts to embed overhead O_2. Then, first embed O_2 data, flag value, and zero point along with their indices. If the peak is not sufficient, then the next peak P_3 and the zero point Z_3 are determined and the remaining overhead is embedded similarly. This process repeats until all overhead information is embedded.

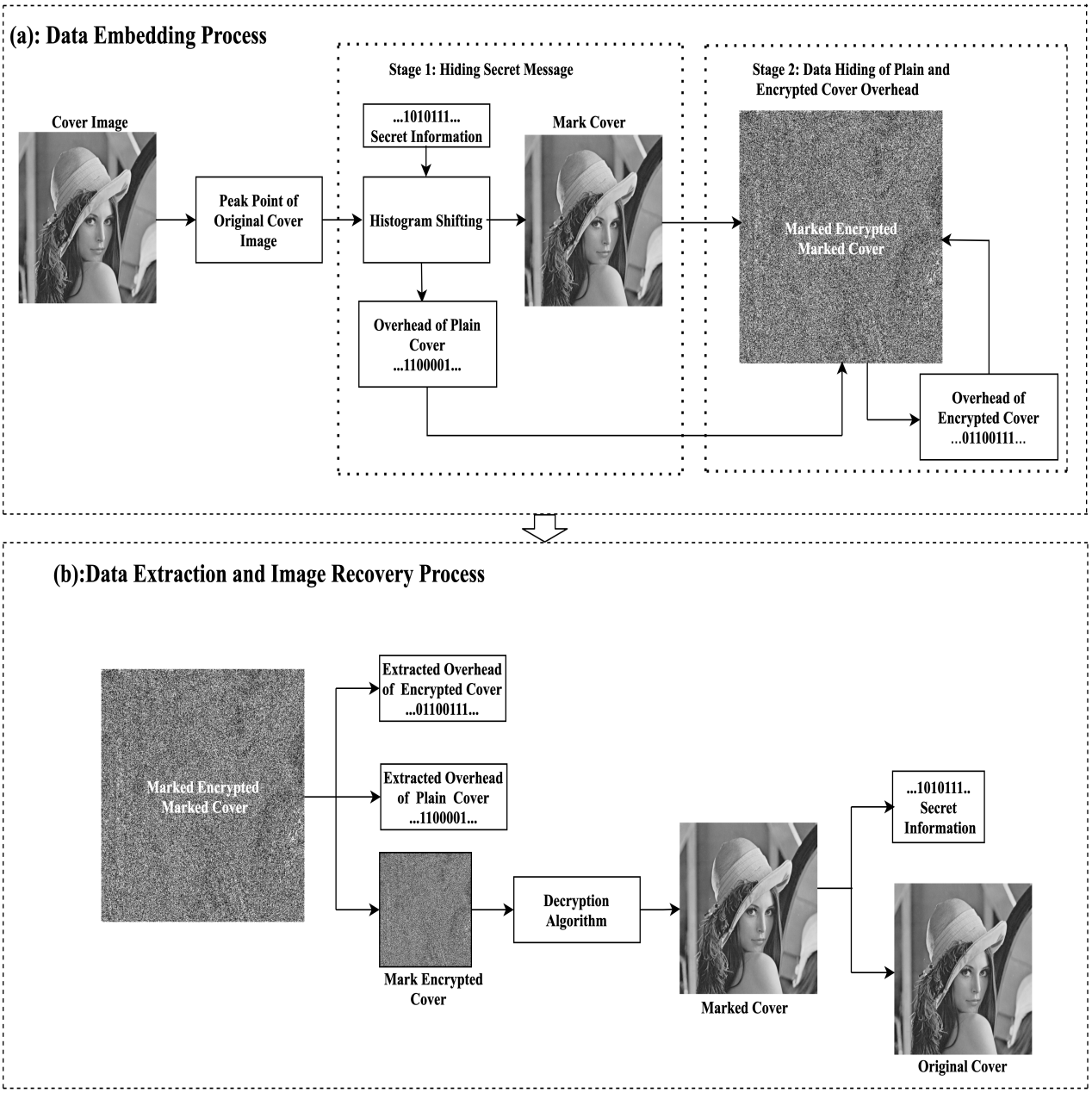

Fig. 3.1 describes embedding process of secret information in plain image and overhead information in encrypted image. (b), shows secret information extraction process along with overhead information and recovery of Original image

3.3.2 Data Extraction and Image Recovery

In the data extraction and image recovery process, the receiver receives a marked encrypted image from the sender, which consists of the secret message and overhead information of the plain and encrypted images. Next, all peaks are determined in descending order in terms of frequency. For each peak P_i, whenever peak pixels P_i and P_i+1 occur, then secret message bits 0 and 1 are extracted respectively. If the extracted data contains a flag value, then, first expected peak is found. If extracted data is consisted only one flag value, then, first 8-bits gives next peak P_2 pixel value and second 8-bits for minimum point Z_2 and from 17^{th} bit upto flag value indicates overflow, whereas 8-bits after flag value represent peak value P_1 and next 8-bits represent minimum point Z_1 and from 17^{th}-bit till the flag gives indices of minimum point. With the help of peak values and overhead information, secret information is extracted along with the original cover image without any errors. In the next subsection, the flowchart of data embedding and extraction is given as follows:

3.3.3 Pseudo code of Proposed method

Pseudo Code for Data Embedding

1. Input: gray-scale image I, number of bits to be embedded per peak pixel(n), binary data to be embedded (D)
2. Generate histogram for I
3. Traverse I to find the maximum point peak P_1 and minimum point (value with zero or minimum frequency) Z_1 .
4. Assumption $P_1 < Z_1$ if Z_1 found, else $Z_1 = 256$

5. Traverse image I to store the plain overhead O_1, by saving the row and column indices of pixels whose value lie between $(Z_1, Z_1 + 2^n - 1)$ or on increasing with $(2^n - 1)$ will become 255.

6. Traverse I and shift pixels with value in (P_1, Z_1) with $(2^n - 1)$.

7. Traverse I and loop through D to get the next n bits that is to be inserted (say K)

   a Convert K to its decimal equivalent

   b If pixel value is equal to P_1, increment pixel value with K.

   c continue traversing and move to the next n bits.

8. Apply affine cipher on marked image I and generated encrypted image I'

9. Traverse to find peak point P_2 and zero point Z_2.

10. Assumption $P_2 < Z_2$ if Z_2 found, else $Z_2 = 256$

11. Traverse I' to store the encrypted overhead O_2 , by saving the row and column indices of pixels whose value lie between $(Z_2, Z_2 + 1)$ or an increasing with 1 will become 255.

12. Traverse I' and shift pixels with value in (P_2, Z_2) with 1 .

13. Traverse I' and loop through O_1 to get the next bit b that is to be inserted:

    a If pixel value is equal to P_2, add one (b).

    b continue traversing and move to the next bit.

14. After embedding O_1 (let the image be (I_J)if the remaining count of is sufficient enough to embed O_2 then From the peak P_2 pixel where no data is embedded, firstly embed the flag, then , then again, the flag and value of Z_2

15. If P_2 is entirely used, then

a Traverse I'' to find the peak P_3 and zero point Z_3

b Assumption $P_3 < Z_3$, if Z_3 found, else $Z_3 = 256$.

c Traverse I_J to store the encrypted overhead O_3, by saving the row and column indices of pixels whose value lie between $(Z_3, Z_3 + 1)$ or on increasing with 1 will become 255.

d Traverse I'' and shift pixels with value in $lbrace P_3, Z_3\}$ with 1.

e Traverse I'' and loop through O_3 to get the next bit (b) that is to be inserted.

i If pixel value is equal to P_3, add bit (b) and continue traversing and move to the next bit.

ii After O_3 is embedded, embed the flag and value of Z_3 and peak P_3

Pseudo Code for Data Extraction and Image Recovery

1. Let the received image I
2. Generate all peaks of I in decreasing order of frequency
3. For each peak P_i ,Traverse the data embedded image (I), if pixel value is in range $(P_i, P_i + 1)$ the data extracted will be 0 if P_i else 1 if $P_i + 1$.
4. Extract data for each peak, if extracted data contains the flag. Then first peak where flag is found, i.e the expected peak P_i.
5. Traverse I, if pixel value is in range $(P_1 + 1, Z_1 + 1)$, decrement the value with 1 to get the image I'.
6. If the extracted data has one flag, then

a First 8-bits give the peak P_2, next 8 has the minimum point Z_2, then from 17^th

bit till the flag gives us the encrypted overflow O_1 and 8 bits after flag gives minimum point Z_1.

b Traverse the image I', if pixel value is in range (P_2, P_2+1), the data extracted will be the difference between pixel value and P_2. Extracted data has first 8 bits of the peak P_3 , next 8 has the minimum point Z_3, then from $17^t h$ bit till the flag gives us the plain overflow O_2 to (a).

c Traverse I', if pixel value is in range (P_2+1, Z_2+1), decrement the value with 1.

d Replace the overhead indices stored in O_1 in the image I'.

7. If the extracted data has two flags, then First 8 bits give the peak , next 8 has the minimum point Z_3, then from $17_t h$ bit till the flag gives us the plain overflow O_2 and bits between the two flags is the encrypted overflow O_1, 8-bits after flag gives minimum point Z_1.

8. Decrypt the Image I' to I''.

9. Traverse the image I'' if the pixel value is in range $[P_3, P_3+(2^n-1)]$, the data extracted will be the binary equivalent of the difference between pixel value and P_3 and extracted data is the secret message.

10. Traverse I'', if pixel value is in range $[P_3, Z_3+(2^n-1)]$,decrement the pixel value with (2^n-1).

11. Replace the overhead indices stored in plain overhead O_2 to get the original image to decrease the value by (2^n-1).

The proposed method is validated on different grayscale images. Details of the experimental results are provided in the next section.

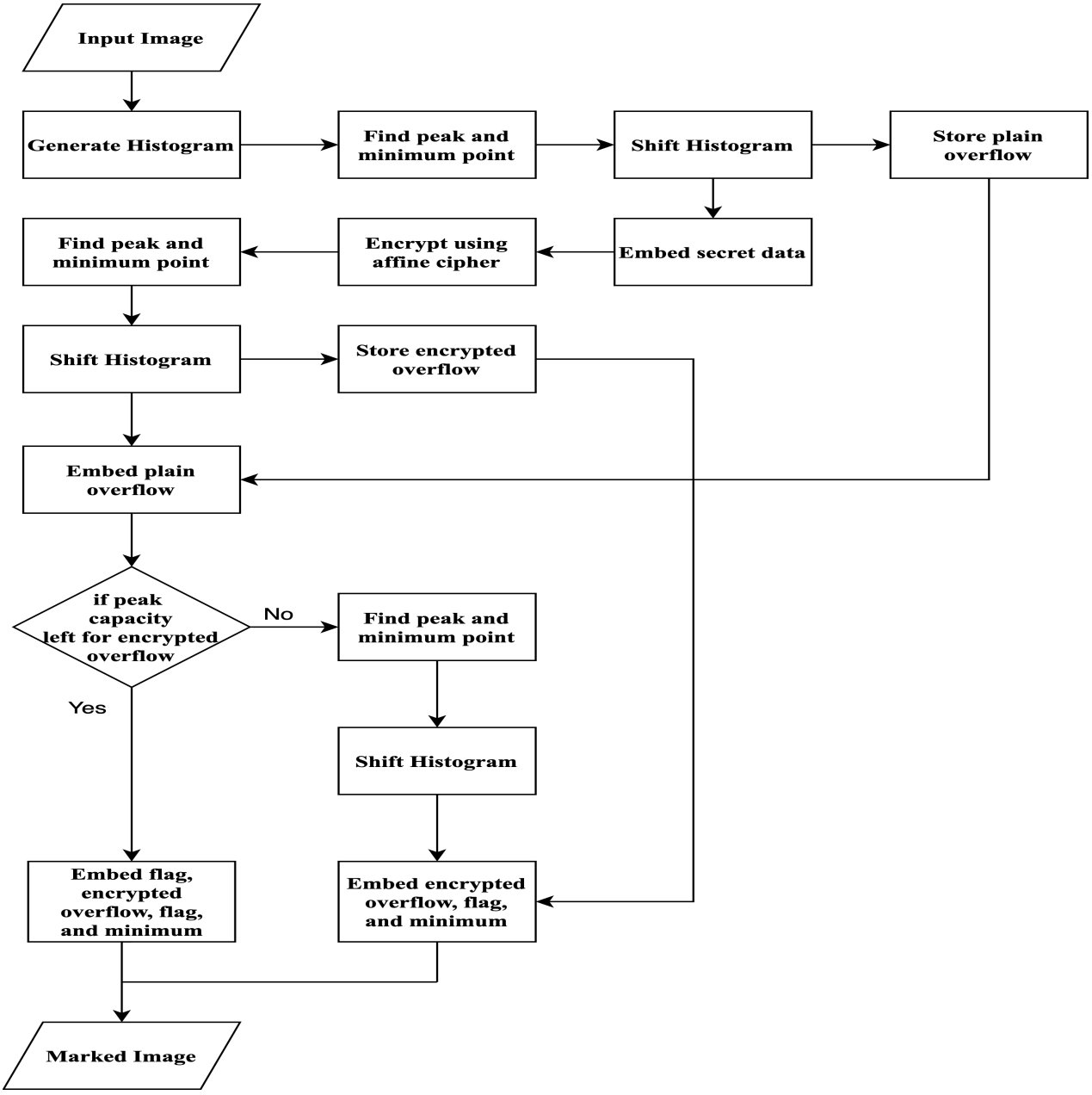

Fig. 3.2 Flow Chart for Data Embedding

3.3.4 Experimental Validation

Experimental validation is performed over different datasets and the results are compared with existing RDH methods. To gauge the robustness of the method, results are obtained over both smooth and texture images, and the significant work reported method is deduced through a detailed analysis and the details of experimental validation following in turn.

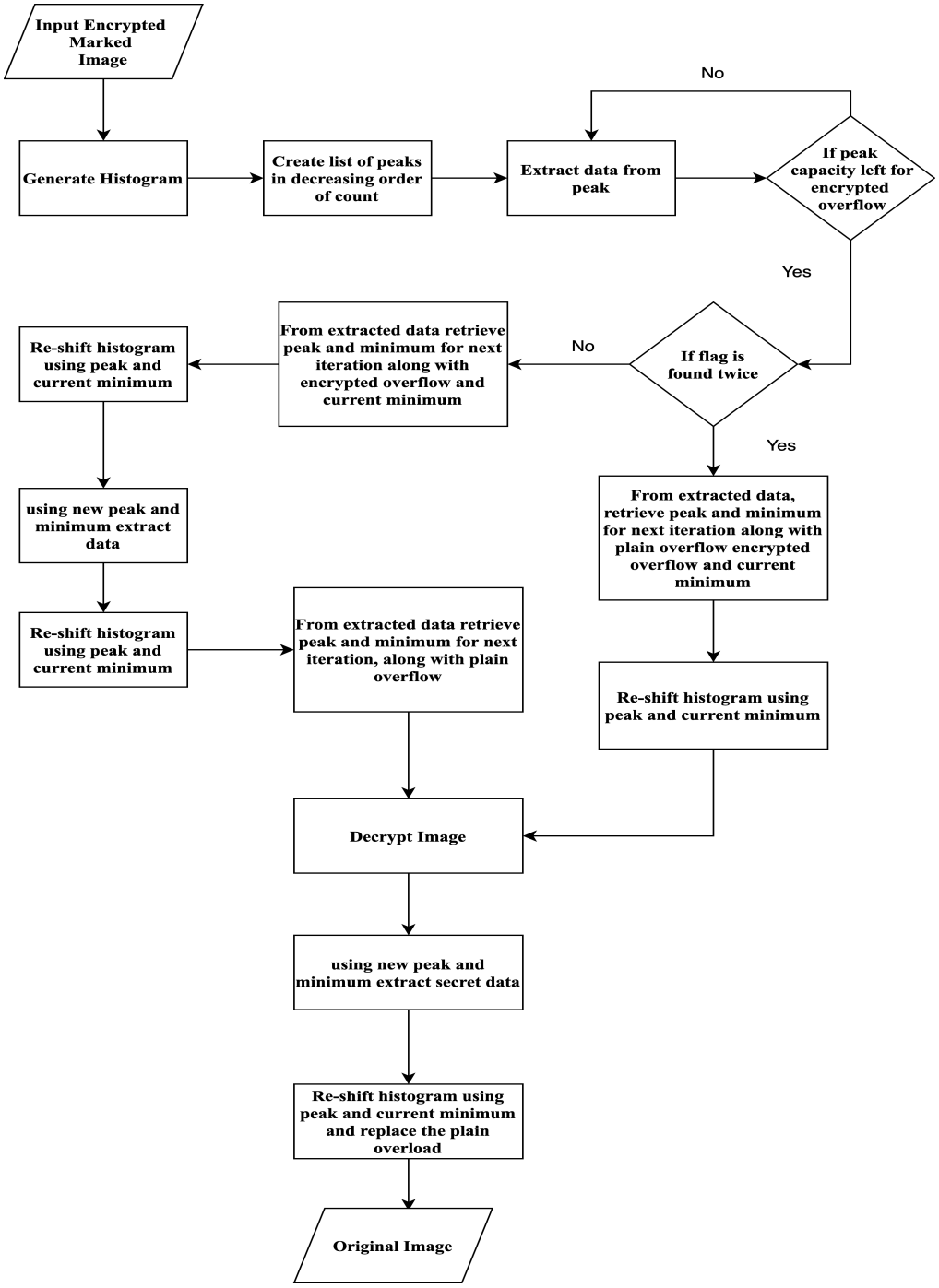

Fig. 3.3 Flow Chart for Data Extraction and Image Recovery

3.3.5 Experimental Set-up

In this section, various observations are conducted to illustrate the performance of the proposed technique. We did experiments on smooth as well as textured grayscale images and found that textured images give sufficient zero pixels to overcome the overflow prob-

lem. It significantly reduces the overhead in comparison to the smooth images. Different datasets are used for smooth and textured images. We considered different block sizes for data hiding with different data bits to enhance the embedding capacity, and corresponding overflow was observed. The secret message bit sequence is generated using pseudo pseudo-random number generator.

The Proposed technique is implemented in the Python programming language and has been tested on a MacBook Air with the following configuration: Apple $M1$ chip with 8 core, RAM 8GB, and Python 3.7. Results are summarized in the following subsection in tabular form.

3.3.6 Results and Analysis for Smooth Images

In the table, we have shown the embedding capacity, PSNR and overheads corresponding to 1, 2, and 3 bits.

Table 3.1: Embedding and overhead results for 1, 2, and 3-bit chunks in plain and encrypted images

Image	**For 1, 2 and 3-bits chunks embedding in Plain Images**									**Encrypted Image**		
512×512	EC_1	PR_1	O_1	EC_2	PR_2	O_2	EC_3	PR_3	O_3	O'_1	O''_2	O''_3
Lena	2711	54.02	16	5422	44.48	16	3133	37.12	16	80	80	96
Baboon	2772	50.45	16	5544	40.90	16	8316	33.50	16	80	80	96
Boat	5796	52.09	16	11592	42.54	94	17388	35.19	458	106	236	1058

Here in Table 3.1, plain/encrypted domain overhead for one-bit embedding is the same for Lena, Baboon, and Boat images, while overhead is the same for Lena and Baboon but different for the Boat image. For two and three bits embedding, only the boat image overhead is different from the other two images. Average embedding capacities are 3760 bits, 7520 bits, and 11280 bits, along with average PSNR of 52.19 dB, 42.64 dB, and 35.27 dB corresponding to one, two, and three bits for Lena, Baboon, and Boat images. The histogram representation of these is shown in Fig. 3.4

In Table 3.2, the block-wise data embedding is done of block size 128×128 and embedding capacity, PSNR, and overhead of plain image & encrypted image are mentioned for one, two, and three bits embedding.

Table 3.2: Data Embedding for 1,2 and 3- bits and Corresponding Overheads for Smooth Images with 128x 128 Blocks

Image	**For 1, 2 and 3-bits chunks embedding in Plain Images**									**Encrypted Image**		
512×512	EC_1	PR_1	O_1	EC_2	PR_2	O_2	EC_3	PR_3	O_3	O'_1	O''_2	O''_3
Lena	4774	50.29	64	9548	40.75	64	14322	33.39	64	384	384	384
Baboon	2961	52.43	64	5922	42.89	64	8883	35.54	64	384	384	1372
Boat	6067	52.01	64	12134	42.47	142	18201	35.12	506	384	592	1320

Overhead for one bit embedding is the same for all three images in plain image and the encrypted image overhead is also the same for all images. For two- and three-bit data embedding, the overhead of the Boat image is more than the overhead of the other two images in the plain image as well as the encrypted image. The average embedding capacity of all three images is 8601 bits, 9201 bits, 13802 bits, corresponding to average PSNRs 51.57 dB, 42.03 dB, 34.68 dB for one, two, and three data hiding bits, respectively. In Table 3.3, we divide the grayscale texture image into blocks of size 64 × 64 blocks. For this, the PSNR of the marked image corresponding to one embedding bit is higher than 50 dB and for two data embedding bits, it lies between 33 dB to 34 dB for the three used images. Overhead for marked Lena and baboon image is the same for the fixed number of data bits, whereas it is different for the Boat image. Overhead for encrypted image is also observed the same.

Table 3.3: Data Embedding for 1,2 and 3- bits and Corresponding Overheads for Smooth Images with 64x 64 Block Size

Image	**For 1, 2 and 3-bits chunks embedding in Plain Images**									**Encrypted Image**		
512×512	EC_1	PR_1	O_1	EC_2	PR_2	O_2	EC_3	PR_3	O_3	O'_1	O''_2	O''_3
Lena	6895	50.87	256	13790	41.34	256	20685	33.99	256	1280	1280	1280
Baboon	3985	50.77	256	7970	41.24	256	11955	33.89	256	1280	1280	1280
Boat	7951	50.67	334	15902	41.14	334	23853	33.79	672	1358	1358	1696

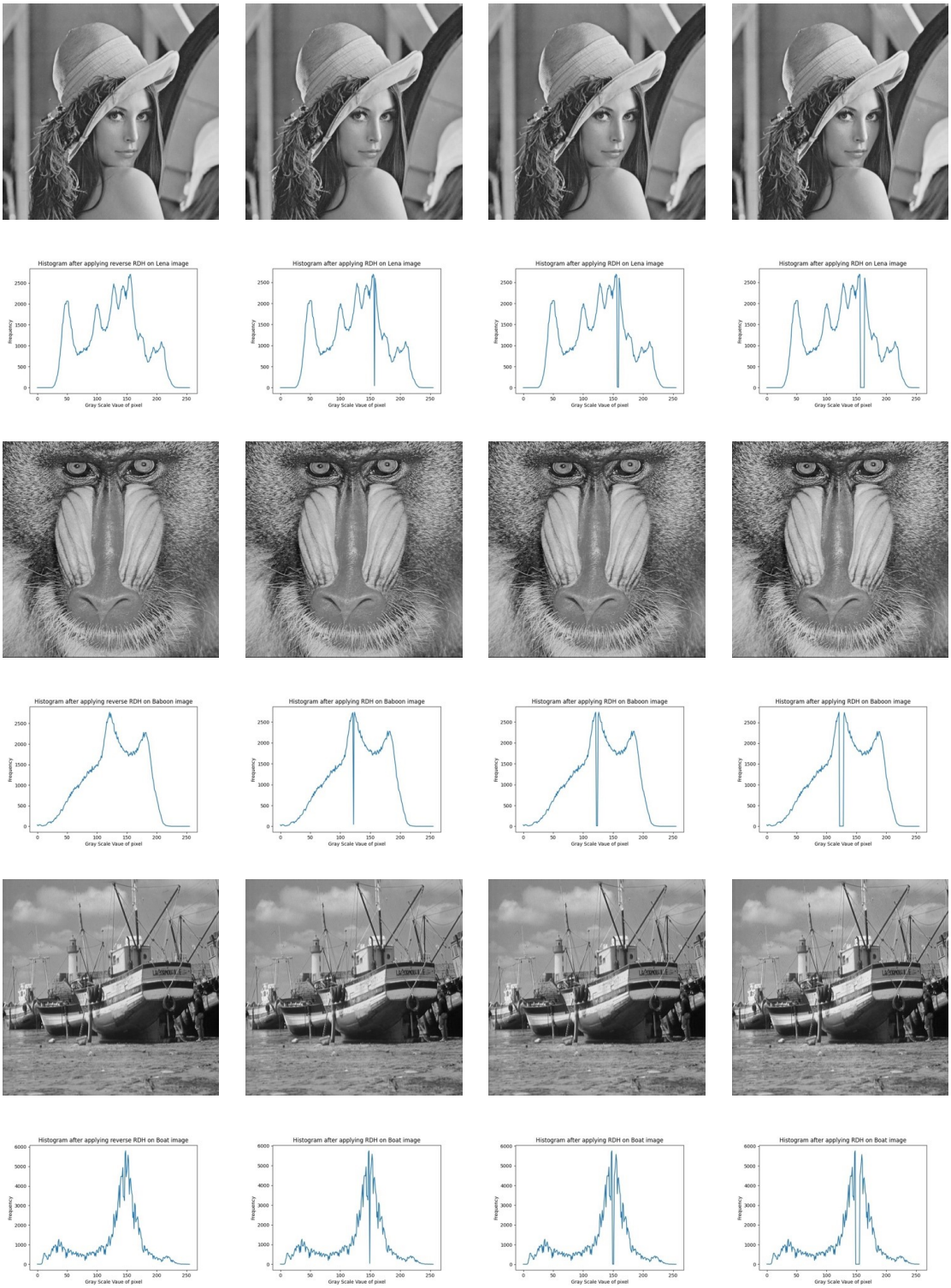

Fig. 3.4 Test image(512x512) set: Lena, Baboon and Boat with Original, 1-bit embedding, 2-bits embedding and 3-bits embedding and corresponding histogram.

3.3.7 Results and Analysis for Texture Images

We have also conducted experiments on different textured images along with different embedding bits and block sizes. Performance analysis of proposed approach in terms of

data hiding capacity, overhead and PSNR are shown in Table 3.4, Table 3.5, and Table 3.7 corresponding to one, two, and three embedding bits. Details of experimental results for the textured image are given in tabular form.

Table 3.4 describes the experimental results for one bit chunks embedding. Twenty-five texture images of size 512 × 512 are used for one-bit embedding, showing the overhead of the marked and encrypted marked image for each image. The overhead of the marked image and the encrypted marked image is the same due to obtaining one zero point in both the images.

Table 3.4: One-bit chunks embedding

Image	**Plain Image**				**Encrypted Image**		
512×512	**Peak Value**	**Pure EC**	**Overhead**	**PSNR (dB)**	**Peak**	**#Peak Count**	**Overhead**
1.1.01	152	2031	16	52.33	26	1986	16
1.1.02	184	3388	16	54.13	11	3305	16
1.1.03	192	3249	16	55.34	11	3238	16
1.1.04	204	4358	16	56.25	108	4292	16
1.1.05	144	4171	16	50.93	157	4118	16
1.1.06	83	3387	16	48.74	8	3304	16
1.1.07	176	4998	16	54.60	13	4936	16
1.1.08	160	8087	16	50.73	99	8004	16
1.1.09	192	6998	16	52.48	67	6915	16
1.1.10	152	3238	16	50.50	43	3155	16
1.1.11	0	13312	16	48.35	3	13229	16
1.1.12	170	4944	16	52.27	26	4925	16
1.1.13	112	2563	16	51.31	118	2518	16
1.2.02	190	3388	16	54.73	181	3305	16
1.2.03	206	3249	16	56.07	196	3238	16
1.2.04	217	4358	16	56.66	248	4292	16
1.2.05	120	4171	16	51.57	30	4118	16
1.2.06	33	3387	16	48.93	42	3304	16
1.2.07	199	4998	16	55.66	43	4936	16
1.2.08	115	8087	16	50.95	232	8004	16

Twenty texture images are used for one bit embedding, two bits chunks embedding and three bits embedding from a publicly available image dataset. Experimental results are

shown in tables, and they depict that the overhead is 16 bits for one, two, and three embedding bits due to sufficient zeros occurring in the texture image. However, it is not always possible to find the zeros i.e $2^n - 1, n >= 2$, in the smooth images.

Table 3.5: Two-bit chunks embedding

Image	**Plain Image**				**Encrypted Image**		
512×512	**Peak Value**	**Pure EC**	**Overhead**	**PSNR (dB)**	**Peak**	**#Peak Count**	**Overhead**
1.1.01	152	2031	16	52.33	26	1986	16
1.1.02	184	3388	16	54.13	11	3305	16
1.1.03	192	3249	16	55.34	11	3238	16
1.1.04	204	4358	16	56.25	108	4292	16
1.1.05	144	4171	16	50.93	157	4118	16
1.1.06	83	3387	16	48.74	8	3304	16
1.1.07	176	4998	16	54.60	13	4936	16
1.1.08	160	8087	16	50.73	99	8004	16
1.1.09	192	6998	16	52.48	67	6915	16
1.1.10	152	3238	16	50.50	43	3155	16
1.1.11	0	13312	16	48.35	3	13229	16
1.1.12	170	4944	16	52.27	26	4925	16
1.1.13	112	2563	16	51.31	118	2518	16
1.2.02	190	3388	16	54.73	181	3305	16
1.2.03	206	3249	16	56.07	196	3238	16
1.2.04	217	4358	16	56.66	248	4292	16
1.2.05	120	4171	16	51.57	30	4118	16
1.2.06	33	3387	16	48.93	42	3304	16
1.2.07	199	4998	16	55.66	43	4936	16
1.2.08	115	8087	16	50.95	232	8004	16

The overhead of the plain and encrypted domains is embedded into the encrypted domain, while the plain domain is used only for secret data embedding. Encrypted domain is only used for the embedding of the overhead information of both domains, and this overhead information is distinguished by the flag value. Plain domain overhead information is embedded before the flag, and encrypted domain overhead information is embedded after the flag value. With the help of this overhead information, the secret message is extracted along with the original cover without any error.

3.3.8 Comparison with State-of-the-art Methods

In this section, a comparison between state-of-the-art methods [4, 30, 45, 103, 184] and our proposed method is carried out and is shown in Table 3.6. In this, our method provided better embedding capacity for both smooth and textured images in comparison to the other RDH methods. However, PSNR is lower than some state-of-the-art methods, but there is no use of PSNR in the encrypted domain. The proposed method is better compared to [4, 30, 45, 103, 184] in terms of data hiding capacity. Details of data hiding capacity and PSNR of our method are given in a separate table for different chunks of bits with their PSNR. In the table, average data hiding capacity and PSNR for block size 128×128 and 64×64 for smooth images and 3-bits for texture images without block are used for comparison with the RDH method.

Table 3.6: Comparison table between state-of-the-art methods and our proposed method

Methods	**Average embedding capacity for 512×512 grayscale image (bits)**	**Average PSNR**
Honsinger [30]	Less than 1024	...
Fridrich [4]	Less than 1024	...
Vleeschouwer [184]	Less than 4096	Less than 3539.0
Zheng [103]	10747	52.3
Arham [45]	4718	44.3
Proposed method for block size of 128×128 for smooth images with three bits embedding	13802	34.68
Proposed method for block size of 64×64 for smooth images with	18831	33.89
Proposed method for three chunks bits of texture images	15962	36.07

Table 3.7: Three-bit chunks embedding

Image	**Plain Image**				**Encrypted Image**		
512×512	**Peak Value**	**Pure EC**	**Overhead**	**PSNR (dB)**	**Peak**	**#Peak Count**	**Overhead**
1.1.01	152	2031	16	52.33	26	1986	16
1.1.02	184	3388	16	54.13	11	3305	16
1.1.03	192	3249	16	55.34	11	3238	16
1.1.04	204	4358	16	56.25	108	4292	16
1.1.05	144	4171	16	50.93	157	4118	16
1.1.06	83	3387	16	48.74	8	3304	16
1.1.07	176	4998	16	54.60	13	4936	16
1.1.08	160	8087	16	50.73	99	8004	16
1.1.09	192	6998	16	52.48	67	6915	16
1.1.10	152	3238	16	50.50	43	3155	16
1.1.11	0	13312	16	48.35	3	13229	16
1.1.12	170	4944	16	52.27	26	4925	16
1.1.13	112	2563	16	51.31	118	2518	16
1.2.02	190	3388	16	54.73	181	3305	16
1.2.03	206	3249	16	56.07	196	3238	16
1.2.04	217	4358	16	56.66	248	4292	16
1.2.05	120	4171	16	51.57	30	4118	16
1.2.06	33	3387	16	48.93	42	3304	16

3.4 Proposed Framework for Bi-directional Secret Message Communication

This section discusses the embedding of the secret message and recovery of the cover image along with the extraction of secret messages in detail. Secret message embedding is depicted in Fig. 3.5, while the cover image recovery process is depicted in Fig. 3.6. Fig. 3.5 consists of three stages, namely preprocessing, secret message embedding, and overhead hiding. In stage 1, the cover image is pre-processed to determine the left and right peaks of the cover image from the main peak. Then, the histogram is shifted in both directions to create the free space for secret message embedding. Stage 2 utilizes both

the peaks and free space, which is determined in the first stage, for secret message embedding. The secret message is embedded into both the peaks, and a marked image is generated. Stage 3 encrypts the marked image, and this encrypted image is preprocessed to determine the left and right peaks of the encrypted image. Then, both peaks are utilized to hide the overhead information of the cover and the encrypted image. Finally, an encrypted marked image is generated and sent to the receiver.

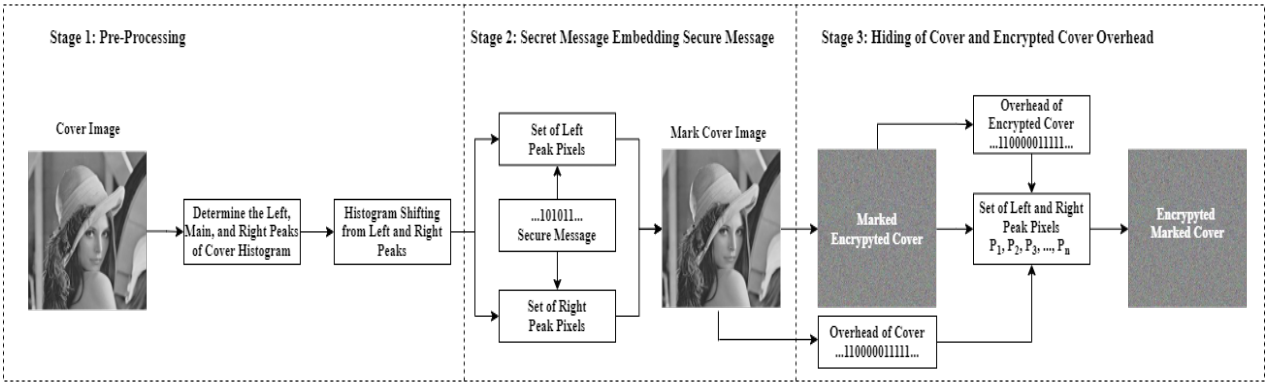

Fig. 3.5 Overview of Secret Data Embedding in Cover Image. (a) stage 1: pre-process the cover image to determine the left and right peaks. (b) stage 2: secret message embedding in plain cover. (c) stage 3: Overhead embedding of plain and encrypted image.

Secret message extraction and cover image recovery process are depicted in Fig. 3.6. In this stage 1, the receiver extracts the overhead information of the cover and encrypted image. For this, both peaks of the encrypted marked image are determined, and using these peaks, overhead information is extracted. The overhead information of the cover and the marked encrypted image is distinguished through the flags. Then, an encrypted image is generated with the help of the overhead of the marked encrypted image. In stage 2, the firstly encrypted image is decrypted using the decryption key, and a marked cover image is generated. Further, overhead information of the cover image is used to extract the secret message and recover the cover image. The proposed technique uses a double layer of data embedding to ensure reversibility. The core design of the proposed methods is given in the subsequent subsections.

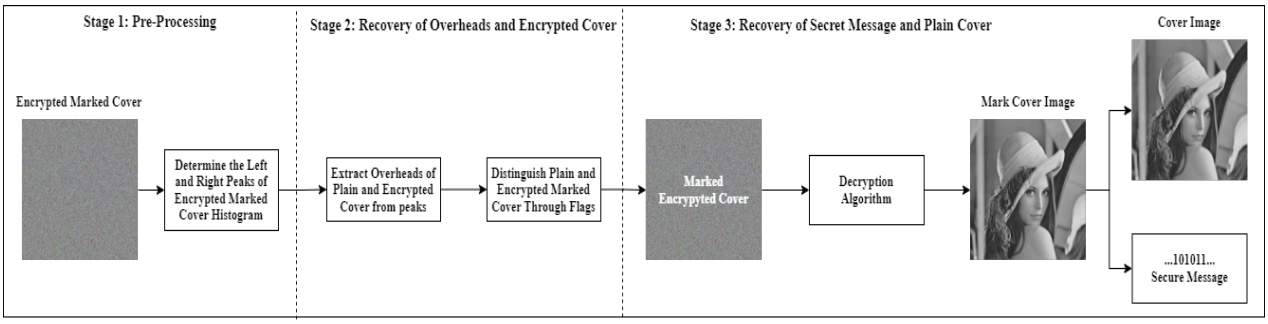

Fig. 3.6 Overview of Secret Data Extraction and Recovery of Cover Image. (a) stage 1: pre-process the encrypted marked image to determine the left and right peaks.(b) stage 2: recovery of overhead of plain and encrypted image. (c) stage 3: secret message extraction and original image recovery.

3.4.1 Core Design of Proposed Bi-directional Data Embedding Technique

The core design of the proposed data embedding phase is depicted as a flow chart in Fig. 3.7. Also intermediate functions for the data embedding algorithm are described from Algorithm 1 to 5 separately. The complete pseudo code for the data embedding technique is summarized as algorithm 6. The details of the data embedding technique follow in turn. Firstly, the main peak p_1 of the cover image is identified and then the cover image is traversed from the left and right sides towards the peak. It locates the next peaks p_1_left and p_1_right and determines the corresponding minimum points z_1_left and z_2_right on both sides. Then, pixels are shifted $2^n - 1$ places after the peak p_1_right up to a minimum point, and pixels are shifted $2^n - 1$ places before the peak p_1_left up to a minimum point because each peak pixel carries n bits of data. Further, secret data is divided into two groups based on the embedding capacity of each peak. Then, clusters of n bits from secret data are selected and converted into their corresponding decimal values. These value are embedded into the bins formed on both sides of the main peak using histogram shifting by adding the secret data value to every pixel with a value equal to the right peak, and subtracting the data value from pixels with value equal to the left peak and accordingly

marked image is generated. Some pixels lead to either overflow or underflow during the pixel shifting. These pixels and their indices are recorded as overhead information. This overhead information is embedded in the second stage i.e, encrypted image. For this, marked image is encrypted using affine cipher to maintain the peak frequency. Then, the encrypted image is further processed in order to embed overhead data in the same way that the original secret bits were embedded. A new largest peak is located in the histogram for embedding, and the left and right peaks that correspond to it are detected. This encrypted version of the cover image hides the overhead bits on the left and right peaks by shifting one place of the histogram in both directions.

This iterative process repeats to embed the current overhead data. If the current peak has sufficient capacity to accommodate the overhead data, it is embedded there; otherwise, a new peak is calculated, and the process repeats until all overheads are embedded into the image. The main functions of the data embedding algorithm are discussed as follows.

(i) hist_plot: The '$hist\_plot$' function takes an image as input and generates a histogram of its pixel intensities. It calculates the histogram using NumPy's 'np.histogram' function, constructs a dictionary mapping pixel values to frequencies, and plots the histogram using Matplotlib. Then, it returns the histogram array. Pseudo code for this is described as algorithm 1.

(ii) format_Data : The function, '$format\_data$', takes a list of secret data and a chunk size *n* as input. It converts the binary data into decimal values by grouping them into chunks of size *n*. If the length of the data list is not divisible by *n*, it pads the data with zeros. Then, it iterates over the padded data list, converting each chunk of binary data into a decimal value and storing it in a new list. Finally, it returns the list of decimal values. Pseudo code for this is described as algorithm 2.

Algorithm 1 Generation of Histogram

```
function HIST_PLOT(image)
    hist, bins ← np.histogram(image, bins=256, range=(0, 256))
    hist_dict ← {}
    for each pixel_value, frequency in ZIP(bins[: −1], hist) do
        hist_dict[int(pixel_value)] ← frequency
    end for
    plot histogram using PLT.HIST(image.ravel(), bins=256, range=(0, 256))
    return hist
end function
```

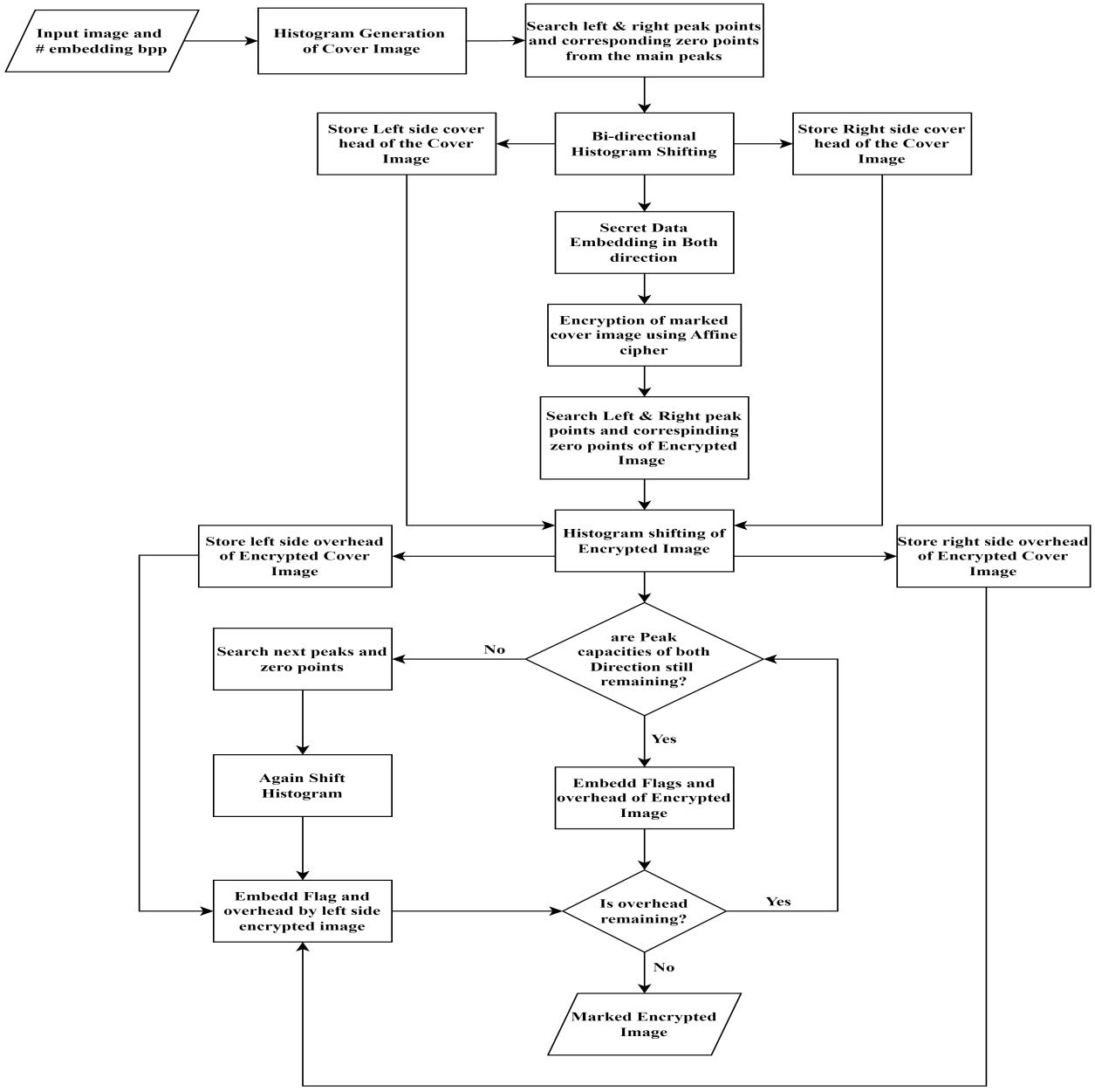

Fig. 3.7 Flowchart of Data Embedding Technique

(iii) find_overflow: The $'find\_overflow'$ function is designed to detect overflow pixels

Algorithm 2 Generation of Formatted Data

```
function FORMAT_DATA(data_list, n)
    dec_arr ← []
    rem ← len(data_list) mod n
    if rem ≠ 0 then
        data_list ← np.concatenate((np.zeros(n − rem, dtype = int), data_list))
    end if
    for i ← 0 to len(data_list) step n do
        chunk ← data_list[i : i + n]
        dec_val ← int("".join(map(str, chunk)), 2)
        dec_arr.append(dec_val)
    end for
    return dec_arr
end function
```

within an image, particularly focusing on those surrounding a specific threshold value *z*. By accepting the image data, *z*, and the number of bits size *n*, the function traverses through the image, identifying pixels that fall within the defined range around *z*. The optional parameter *left* determines the direction of the search, allowing flexibility in the overflow detection process. After identifying the pixels within the specified range, the function converts these values into the binary system along with their corresponding row and column indices. These binary values are concatenated into a single list, *o*, which accumulates the overhead data of the detected overflow pixels. This data provides insights into the spatial distribution of overflow within the image, aiding in subsequent processing or analysis. Once the overhead data is calculated, the function prints it along with its length, providing a summary of the detected overflow within the image. The overhead data is represented in binary format and contains essential information about the spatial positions and pixel values of the identified overflow for the extraction of the secure message and recovery of the original cover image. Pseudo code for this is described as algorithm 3.

(iv) mark_image : This $'mark\_image'$ function alters an original cover image by embedding the secure data and generates the marked cover image. It accepts parameters including the cover image, a list of secret messages to be embedded into the cover image,

Algorithm 3 Determine Overflow

```
1:  function FIND_OVERFLOW(image, z, n, left = 0)
2:      o ← []
3:      if left = 1 then
4:          values ← list(range(z − (2^n − 1) + 1, z + 1))
5:      else
6:          values ← list(range(z, z + (2^n − 1)))
7:      end if
8:      for i ← 0 to image.shape[0] do
9:          for j ← 0 to image.shape[1] do
10:             if image[i][j] ∈ values then
11:                 shape ← ⌈log2(image.shape[0])⌉
12:                 pix ← conv_bin(image[i][j], 8)
13:                 col ← conv_bin(j, shape)
14:                 row ← conv_bin(i, shape)
15:                 o.extend(pix + row + col)
16:             end if
17:         end for
18:     end for
19:     return o
20: end function
```

a pixel value pattern (*p*), and direction (*dirn*) which indicates whether the secret message to be embedded to the right or left direction. An optional parameter *start_index* determines the starting point for the marking process within the image and iterates through the image pixels beginning from the *start_index*. For each pixel matching the pattern *p*, the corresponding data value from *data_list* is retrieved and the modification is applied to the pixel value. The data embedding direction specified by *dirn* determines whether the data value is added to or subtracted from the pixel value. This function maintains a pointer (*ptr*) to track the current position within the *data_list* and records the index of the last marked pixel as *last_marked_pixel*. This ensures proper tracking and control over the data embedding process. Once secure data hiding is complete then it returns the index of the last marked pixel for referencing further processing. Pseudo code for this is described as algorithm 4.

(v) affine_cipher_image : Function $'affine\_cipher\_image'$ is applied on the marked cover image to generate the encrypted marked image. It utilizes parameters *a*, *b*, and *m*

Algorithm 4 Generation of Mark Image

```
1:  function MARK_IMAGE(image, data_list, p, dirn, start_index = 0)
2:      assert "Input image must be flattened"
3:      ptr ← 0
4:      last_marked_pixel ← 0
5:      for i ← start_index to image.shape[0] do
6:          if image[i] == p then
7:              k ← data_list[ptr]
8:              image[i] += (k × dirn)
9:              last_marked_pixel ← i
10:             ptr += 1
11:             if ptr ≥ len(data_list) then
12:                 break
13:             end if
14:         end if
15:     end for
16:     return last_marked_pixel
17: end function
```

for generating the marked pixel p'_i of the pixel p_i using the 3.4.1

$$p'_i = (a \times p_i + b)\%m \tag{3.4.1}$$

where *a* and *b* are co-prime integers and *m* is the modulus. This equation modifies each pixel value by multiplying by *a*, adding *b*, and then taking the modulo *m*. This process encrypts the image by preserving its grayscale structure. Pseudo code for this is described as algorithm 5.

Algorithm 5 Generation of Encrypted Marked Image

```
1: function AFFINE_CIPHER_IMAGE(image, a, b, m)
2:     encrypted_image ← (a × p_i + b)  mod m
3:     encrypted_image ← np.uint8(encrypted_image)
4:     return encrypted_image
5: end function
```

3.4.2 Core Design of Bi-directional Data Extraction and Image Recovery Technique

The data extraction and image recovery technique is designed to faithfully recover both the cover and secret message at the receiving end. It consists of three stages, namely pre-processing, recovery of plain overhead & encrypted marked cover, and recovery of secret message and plain cover. The core design of data extraction and plain cover recovery phase is depicted as a flow chart in Fig. 3.8. Intermediate functions for data extraction and image recovery are detailed in the form of algorithms from 7 to 9 separately. The complete pseudo code for data extraction and cover image recovery is given as in algorithm 10. The main functions of data extraction and image recovery are discussed in turn as follows.

The data extraction process starts with the sorting of all peaks based on the histogram in descending order of the encrypted marked cover image. Then, data retrieval commences from each peak in sequential order until a peak containing two flags is found, which provides the information about the computation of data extraction. Data extractor reaches the particular peak containing two flags, leading to the identification of the starting point. This provided the beginning point of the data extraction and holding the data regarding the previous peak, ("p_{prev}" and "z_{prev}") as well as previous overhead values (o_{prev}) preceding the first flag and current peaks, zero point z and overhead values. Embedding changes are reversed by using the data from this peak to return the pixel values to their initial places. After making this modification, the extraction procedure proceeds to the peak that came before it in the sequence. Then, data retrieval resumes and checks the presence of the flag from the next peak. If a flag exists, then it indicates that the previous stage has been reached. This data provides the information of peak, zero point z, and the overhead of the second stage. Then, shift the pixel values to achieve the preceding peak and repeat this

process iteratively until no flag is found. This iterative process necessitates bidirectional execution to ensure the faithful extraction of overhead in both the left and right ends of the image. The encrypted marked cover image is recovered with the help of this overhead information. Then, the receiver decrypts this encrypted image using the decryption key.

Post-decryption of the encrypted image reveals a histogram with embedded data. Secure data is extracted from the left and right peaks of the marked cover image with the help of peaks recovered from the overhead information. Further, pixel values of both sides are shifted according to the data embedding bits, and overflow pixel values are replaced with the original positions using the overhead information. This process ensures exact recovery of the secret message along with the original cover image. The flow chart and main functions for secret message extraction and plain cover recovery are given in turn as follows.

(i) affine_decipher_image : Function $'affine\_decipher\_image'$ decrypts an encrypted

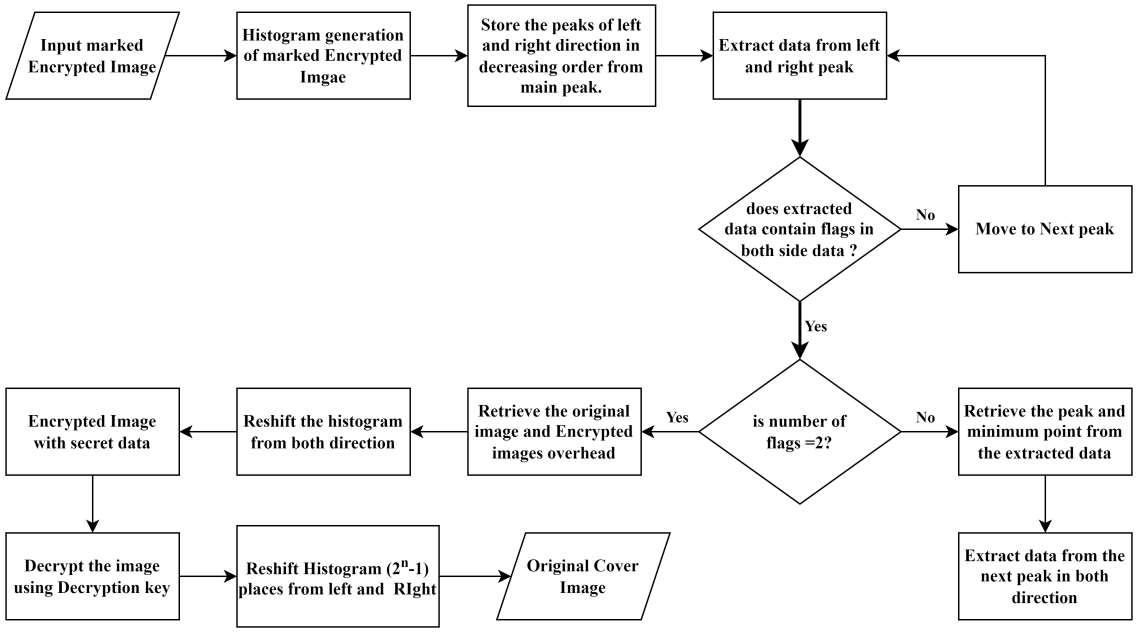

Fig. 3.8 Flowchart for Data Extraction Technique

image. For this, it calculates the modular inverse of the coefficient *a* using the *pow* function to ensure decryption accuracy within the modulus *m*. By using the inverse of *a_inv*

Algorithm 6 Proposed Data Embedding Technique

Input: Original Cover Image and No. of Embedding bits per peak pixel (n)
Output: Encrypted marked image
function $hist\_plot(image)1$
function $format\_data(datalist, n)2$
function $find\_overflow(image, z, n, left = 0)3$
function $Mark\_image(image, datalist, p, dirn, startindex = 0)4$
function $sffine\_cipher\_image(image, a, b, m)5$
$p_2 \leftarrow$ Max_peak_of($hist$)
$p_2\_left, z_2\_left \leftarrow$ Max_peak_left_of_p2($hist, n, 1$), Min_val_left_of_p($hist, n, 1$)
$p_2\_right, z_2\_right \leftarrow$ Max_peak_right_of_p2($hist, n, 1$), Min_val_right_of_p($hist, n, 1$)
$data\_next\_left \leftarrow$ Convert_to_byte_stream($p_1\_left, z_1\_left, o1\_left$, flag)
$data\_next\_right \leftarrow$ Convert_to_byte_stream($p_1\_right, z_1\_right, o1\_right$, flag)
$o_2\_left, o_2\_right \leftarrow$ Compute_Overhead($img, 1$)
$p_{\text{prev left}} \leftarrow p_{1\text{left}}$
$z_{\text{prev left}} \leftarrow z_{1\text{left}}$
$o_{\text{prev left}} \leftarrow o_{1\text{left}}$
$p_{\text{curr left}} \leftarrow p_{2\text{left}}$
$z_{\text{curr left}} \leftarrow z_{2\text{left}}$
$o_{\text{curr left}} \leftarrow o_{2\text{left}}$
while ($data\_next\_left$ is not empty) **do**
    if $data\_next\_left$ fits in current left peak **then**
        $img \leftarrow$ Embed_Data($img, p_{\text{curr left}}, z_{\text{curr left}}, n, data\_next\_left, left = 1$)
        $data\_next\_left \leftarrow$ Convert_to_byte_stream($p_{\text{curr left}}, z_{\text{curr left}}, o_{\text{curr left}}$, flag)
    else
        $p_{\text{prev left}} \leftarrow p_{\text{curr left}}$
        $z_{\text{prev left}} \leftarrow z_{\text{curr left}}$
        $o_{\text{prev left}} \leftarrow o_{\text{curr left}}$
        $p_{\text{curr}} \leftarrow$ Max_peak_of($hist$)
        $p_{\text{curr left}}, z_{\text{curr left}} \leftarrow$ Max_peak_left_of_p_curr($hist, n, 1$), Min_val_left_of_p($hist, n, 1$)
        $o_{\text{curr left}} \leftarrow$ Compute_Overhead(img, n)
        $img \leftarrow$ Shift pixel left$range(z_{\text{curr left}}, p_{\text{curr left}})$ by 1
        $data\_next\_left \leftarrow$ Convert_to_byte_stream($p_{\text{prev left}}, z_{\text{prev left}}, o_{\text{prev left}}$, flag)
        $img \leftarrow$ Embed_Data($img, p_{\text{curr left}}, z_{\text{curr left}}, n, data\_next\_left, left = 1$)
        $data\_next\_left \leftarrow$ Convert_to_byte_stream($p_{\text{curr left}}, z_{\text{curr left}}, o_{\text{curr left}}$, flag)
    end if
end while
Perform same while loop for right.
Output: $marked\_image$

the cover image is recovered using Eq. 3.4.2

$$p_i = (a\_inv \times (encrypted\_p_i - b))\%m \tag{3.4.2}$$

This applies to each pixel of the encrypted image. Finally, the function returns the de-

crypted image for further processing. Pseudo code for this is described in algorithm 7.

Algorithm 7 Affine Decipher Image Decryption

```
function AFFINE_DECIPHER_IMAGE(encrypted_image, a, b, m)
    a_inv ← pow(a, −1, m)
    decrypted_image ← (a_inv × (encrypted_image − b)) mod m
    decrypted_image ← np.uint8(decrypted_image)
    return decrypted_image
end function
```

(ii) count_p : The $'count\_p'$ function tallies the occurrences of a specific pixel value p within an image, starting from a designated index. It iterates through the image, incrementing a counter for each encountered instance of p. An optional parameter allows skipping the first pixel when counting, useful for resuming counting from a specific index. After traversal, it prints the total count of "p" occurrences and returns this count for further analysis. This function is mainly employed to calculate the peak values. Pseudo code for this is described as algorithm 8.

(iii) check_flag: The function $'check\_flag'$ searches for a specific pattern within an array

Algorithm 8 Determine Count Pixels

```
function COUNT_P(image, p, start_index = 0)
    count_p2 ← 0
    skip_first_pixel ← start_index ≠ 0
    for i ← start_index to image.shape[0] do
        if skip_first_pixel then
            skip_first_pixel ← False
            continue
        end if
        if image[i] == p then
            count_p2 += 1
        end if
    end for
    return count_p2
end function
```

and returns the positions where the pattern is found. It iterates through the array, comparing substrings of the same length as the pattern with the pattern itself. If a match is found, the function records the starting position of the match. Finally, it returns a list of

positions where the pattern occurs in the array. Pseudo code for this is described as Algorithm 9. The procedure can also be used on images by dividing them into several blocks

Algorithm 9 Check Flag

```
1:  function CHECK_FLAG(pattern, array)
2:      positions ← []
3:      pattern_str ← ''.join(map(str, pattern))
4:      array_str ← ''.join(map(str, array))
5:      pattern_length ← len(pattern)
6:      for i ← 0 to len(array) − pattern_length + 1 do
7:          if array_str[i : i + pattern_length] == pattern_str then
8:              positions.append(i)
9:          end if
10:     end for
11:     return positions
12: end function
```

of the same size, where N_b is the total number of blocks. Comparing this partition to a single, whole image can either retain or improve the embedding capability. Experimental observations show that dividing the image in fact lowers the number of overhead bits, even if doing so typically exacerbates the problem of low peaks in overhead embedding. However, the likelihood of running across errors during the embedding process also grows as N_b increases.

The method for working with blocks is very straightforward: the image is split into N_b blocks, and each of these is fed separately to the algorithm with the necessary data. Eventually, all the blocks are combined into a single image to create the marked image. At the receiving end, the blocks are split up again appropriately and fed individually to the retrieval algorithm. Finally, the altered image is created by combining all the processed blocks and retrieving the data from each one. The proposed technique is validated experimentally, and details follow in the next section.

Algorithm 10 Proposed Data Extraction Technique

Input: Encrypted Marked Image (img_rec)
Output: Original Cover Image, Secure Data
Set of all pixels in descending frequency order.
for all p in $sorted\_pixels$ **do**
    Extract overhead data from peaks p and $p+1$ and append.
    if len($flag\_found$) = 2 **then**
        $data\_pixels\_right$.append($data\_exc$)
        $flag\_found\_total\_right$.append($flag\_found$)
    end if
    Extract overhead data from peaks p and $p-1$ and append.
    Perform the same process for the left direction.
    Data before first flag represent the overhead of original cover image and from first flag to second flag represent the overhead of Encrypted marked image.
end for
for all pixel in img_rec **do**
    if img_rec in range $p\_prev\_right \leq img\_rec[i][j] \leq p\_prev\_right + 2^n - 1$ **then**
        Extract data from pixels as d
        $data\_r$.append(d)
        $img\_rec[i][j] \leftarrow p\_prev\_right$
    end if
    if img_rec in range $p\_prev\_left \leq img\_rec[i][j] \leq p\_prev\_left + 2^n - 1$ **then**
        Extract data from pixels as d
        $data\_l$.append(d)
        $img\_rec[i][j] \leftarrow p\_prev\_left$

Algorithm 10 Proposed Data Extraction Technique (Continue...)

$p\_prev\_right + 2^n \leq img\_rec[i][j] \leq z\_prev\_right + 2^n - 1$
$img\_rec[i][j] \leftarrow img\_rec[i][j] - (2^n - 1)$
$z\_prev\_left - (2^n - 1) \leq img\_rec[i][j] \leq p\_prev\_left - (2^n)$
$img\_rec[i][j] \leftarrow img\_rec[i][j] + (2^n - 1)$

if (Overhead is present in $o\_prev\_right\_$) **then**
    Mark all the values in img_rec
end if
if len($o\_prev\_left\_$) > 0 **then**
    for all pix, val in $o\_prev\_left\_$.items() **do**
        $i, j \leftarrow val$
        $img\_rec[i][j] \leftarrow pix$
    end for
end if
$data\_bin\_l \leftarrow$ (convert_to_binary($data\_l, n$))
$data\_bin\_r \leftarrow$ (convert_to_binary($data\_r, n$))
Return img_rec, $data\_bin\_l$, and $data\_bin\_r$ =0

3.5 Experimental Validation

We discussed the experimental results of the proposed technique along with different performance metrics. Experimental results validation involves three parts such as experimental design, qualitative validation, and quantitative validation. The details of the experiments are discussed in the subsequent subsection.

3.5.1 Complexity Analysis

The proposed method works similar to one directional histogram shifting. In normal histogram shifting based embedding, data is embedded only on the single peak. Whereas in the proposed method, the peak point is found only for the purpose of determining the second largest left and right peaks, thereby enhancing the embedding capacity of the proposed method. Therefore, traversal will be the same as that of the normal histogram shifting based embedding. The complexity of normal histogram based embedding is $O(MN)$ where $M \times N$ being the size of the image. But multistage embedding is performed in the proposed method, wherein secret data and overhead information are embedded in different stages, each stage of complexity $O(MN)$. Thus, the overall complexity of the proposed method is $K \times O(MN)$ where K is constant.

3.5.2 Qualitative Validation

Qualitative validation of the cover image and its histogram after shifting for different embedding bits is examined. For this, six cover images are considered, and these are depicted in figures from Figs. 3.9 to 3.17 along with their histograms. In each of these figures, (a) represents the histogram of cover image while (b), (c), and (d) show the change

in histograms after shifting the one, three and seven places respectively, corresponding to the one, two and three secret bits embedding in both the directions. In the figures from Figs. 3.9 to 3.17, shifting of one place shows very low distortion in their histograms, and similarly, in Figs. 3.9 to 3.11, the histograms of Lena and Forest for two bits embedding also exhibit very less distortion. However, histograms for three secret message bits embedding have shown more distortion when compared to the one and two bits embedding, even though they offer tunable quality.

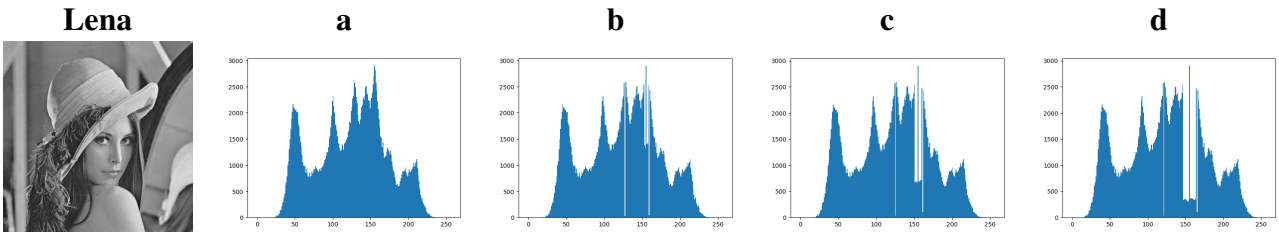

Fig. 3.9 Lena plain cover. (a) histogram of lena plain cover. (b) histogram after one place shifting for one bit embedding. (c) histogram after three place shifting for two bit embedding. (d) histogram shifting after seven place shifting for three bit embedding.

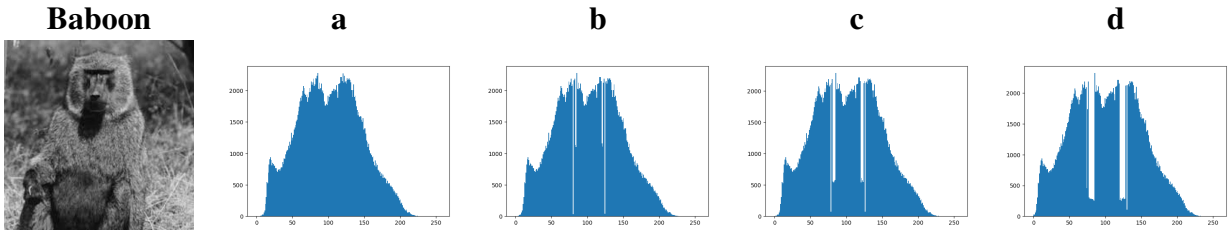

Fig. 3.10 Baboon plain cover. (a) histogram of baboon plain cover. (b) histogram after one place shifting for one bit embedding. (c) histogram after three place shifting for two bit embedding. (d) histogram shifting after seven place shifting for three bit embedding.

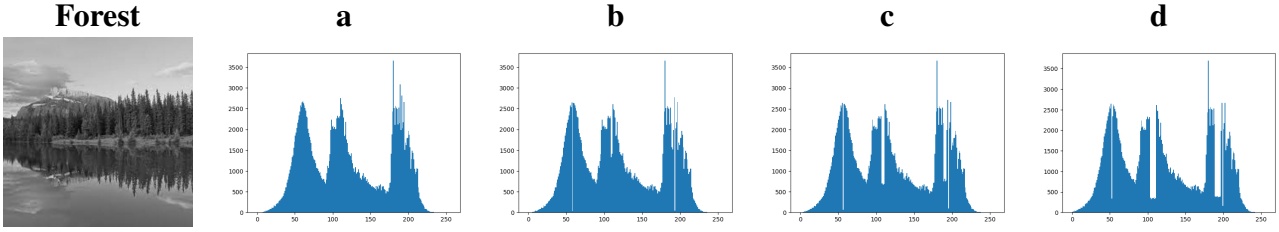

Fig. 3.11 Forest plain cover. (a) histogram of forest plain cover. (b) histogram after one place shifting for one bit embedding. (c) histogram after three place shifting for two bit embedding. (d) histogram shifting after seven place shifting for three bit embedding.

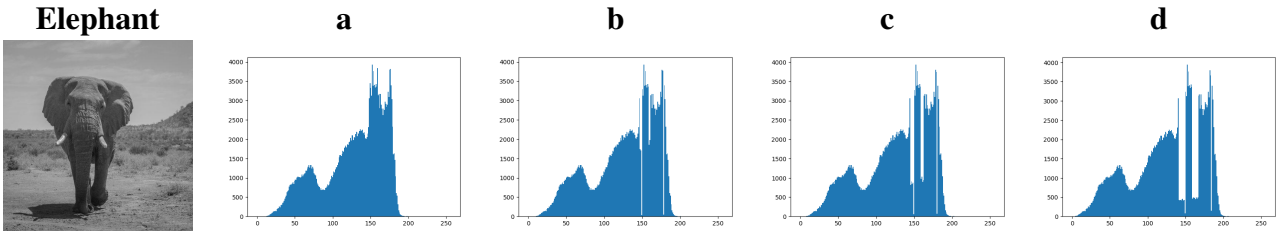

Fig. 3.12 Elephant plain cover. (a) histogram of elephant plain cover. (b) histogram after one place shifting for one bit embedding. (c) histogram after three place shifting for two bit embedding. (d) histogram shifting after seven place shifting for three bit embedding.

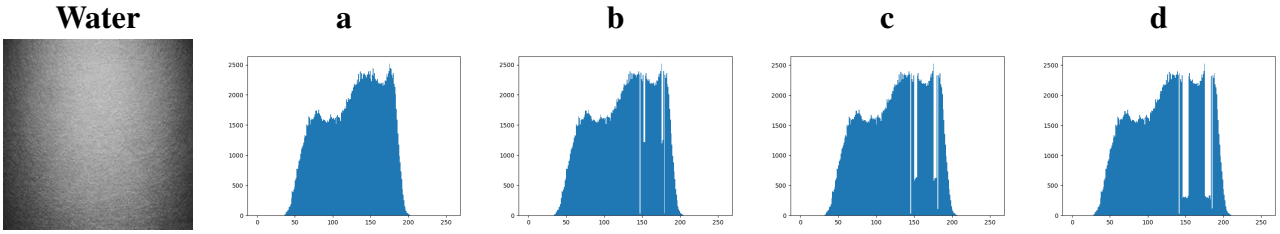

Fig. 3.13 Water plain cover. (a) histogram of water plain cover. (b) histogram after one place shifting for one bit embedding. (c) histogram after three place shifting for two bit embedding. (d) histogram shifting after seven place shifting for three bit embedding.

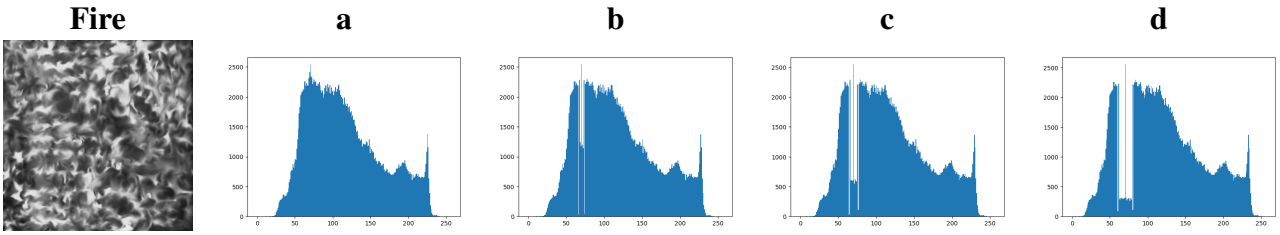

Fig. 3.14 Fire plain cover. (a) histogram of fire plain cover. (b) histogram after one place shifting for one bit embedding. (c) histogram after three place shifting for two bit embedding. (d) histogram shifting after seven place shifting for three bit embedding.

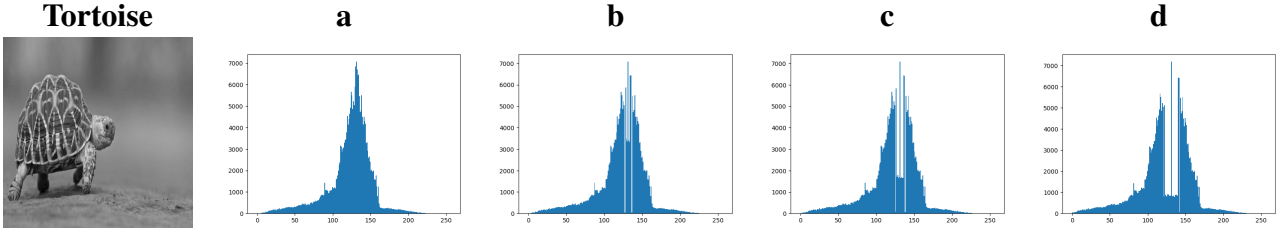

Fig. 3.15 Tortoise plain cover. (a) histogram of tortoise plain cover. (b) histogram after one place shifting for one bit embedding. (c) histogram after three place shifting for two bit embedding. (d) histogram shifting after seven place shifting for three bit embedding.

The proposed technique has a tunable embedding capacity, wherein the per-pixel embedding capacity can be determined according to the amount of secret data. Quantitative validation is discussed for different performance metrics in the next section.

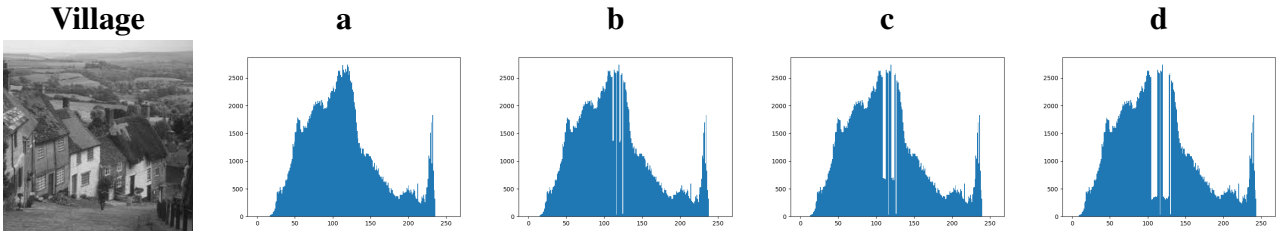

Fig. 3.16 Village plain cover. (a) histogram of village plain cover. (b) histogram after one place shifting for one bit embedding. (c) histogram after three place shifting for two bit embedding. (d) histogram shifting after seven place shifting for three bit embedding.

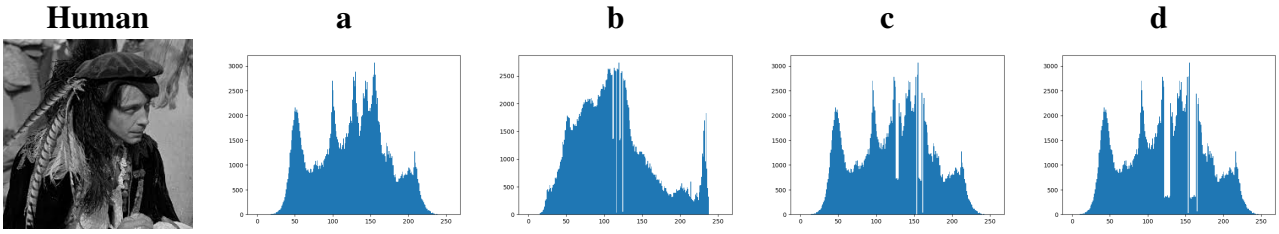

Fig. 3.17 Human plain cover. (a) histogram of human plain cover. (b) histogram after one place shifting for one bit embedding. (c) histogram after three place shifting for two bit embedding. (d) histogram shifting after seven place shifting for three bit embedding.

3.5.3 Quantitative Validation of Proposed Technique

Quantitative analysis of the proposed method is performed over more than 40 8-bit grayscale images of size 512×512. The evaluation revealed that the reconstruction of images and the retrieval of data are accurate at the receiver's side. The number of overhead bits is proportional to the embedding message bits per peak pixel. For instance, as the number of secure message bits per peak pixel is increased, the number of overhead bits is also increased due to the skewed nature of the cover image histogram. The result is that the critical role of image selection ensures the smooth functionality of the method.

In Tables 3.8, 3.9, and 3.10, the embedding capacity along with the PSNR, SSIM, NCC, and POSP, and overhead bits are tabulated. These three Tables include the results for nine different types of grayscale images of size 512×512 without dividing them into blocks, i.e.$N_b = 1$ taking 1-bit, 2-bit, and 3-bit per peak pixel embedding at a time, and corresponding overhead information bits are also summarized. It shows that though the input cover

Table 3.8: Performance Metrics for Different Images of Size 512×512 without Block and (n = 1) bits/peak pixel

Images 512×512	Maximum capacity(bits)	PSNR	SSIM	NCC	POSP	Overheads(bits)
Lena	5353	42.300	0.9982	0.99983	0.9875	16
Babboon	4504	42.8377	0.9984	0.99989	0.9874	16
Forest	5569	43.9797	0.9992	0.99996	0.9756	16
Elephant	7286	43.5606	0.9988	0.99975	0.9817	16
Water	4971	43.3995	0.9984	0.99978	0.9876	16
Fire	5294	43.6438	0.9988	0.99971	0.9903	16
Tortoise	12754	42.8510	0.9974	0.99899	0.9712	16
Village	5431	42.47	0.9976	0.9997	0.9886	16
Human	5707	43.0	0.9988	0.9998	0.9871	16
Average	**6533**	**43.22**	**0.9984**	**0.9997**	**0.9286**	**16**

Table 3.9: Performance Metrics for Different Images of Size 512×512 without Block and (n = 2) bits/peak pixel

Images 512×512	Maximum capacity(bits)	PSNR	SSIM	NCC	POSP	Overheads(bits)
Lena	10706	36.2372	0.9929	0.98150	0.9873	42
babboon	9808	37.6616	0.9933	0.99949	0.9870	120
Forest	11138	37.7542	0.9965	0.99974	0.9749	42
Elephant	14572	37.1160	0.9948	0.99880	0.9814	120
Water	9942	36.8615	0.9929	0.99912	0.9873	16
Fire	10588	37.1268	0.9950	0.99877	0.9901	16
Tortoise	25508	36.6833	0.9899	0.99614	0.9709	16
Village	10862	36.45	0.9913	0.9991	0.9886	16
Human	11414	37.16	0.9956	0.9994	0.9871	42
Average	**13810**	**37.0629**	**0.9936**	**0.99622**	**0.9827**	**53**

Table 3.10: Performance Metrics for Different Images of Size 512×512 without Block and (n = 3) bits/peak pixel

Images 512×512	Maximum capacity(bits)	PSNR	SSIM	NCC	POSP	Overheads(bits)
Lena	16059	30.1952	0.9759	0.99659	0.9873	94
Babboon	13512	31.3997	0.9731	0.99801	0.9842	770
Forest	16707	31.4998	0.9847	0.99868	0.9718	744
Elephant	21858	30.9023	0.9799	0.99539	0.9809	354
Water	14913	30.5341	0.9741	0.99669	0.9872	16
Fire	15882	30.7876	0.9814	0.99561	0.9898	94
Tortoise	38262	30.2716	0.9637	0.98369	0.9657	848
Village	16293	30.42	0.9701	0.9969	0.9886	16
Human	17121	31.20	0.9837	0.9979	9871	26
Average	**19599**	**30.7986**	**0.9851**	**0.99495**	**0.9809**	**417.14**

image is of the same size for each case still the embedding capacity varies due to the varying number of peak pixels. PSNR and SSIM in plain cover image remain in a good range, ensuring that it is visually not very likely to point out that it has been changed. Also, it can be further noticed that when 1 bit per peak pixel embedding is performed, then very less overhead information is required. In Table 3.11, the secure data embedding capacity

Table 3.11: Embedding Capacity for Different Images for blocks of size 256×256

Images 512×512	n=1-bit/peak pixel	n=2- bit/peak pixel	n=3-bit/peak pixel
Lena	9795	19590	29385
Baboon	5562	11124	16686
Forest	13310	26620	39930
Elephant	11194	22388	33582
Water	7007	14014	21021
Fire	5127	10254	15381
Tortoise	15111	30222	45333
Average	**9586**	**19173**	**28760**

Table 3.12: Embedding Capacity for Different Images for blocks of size 32×32

Images 512×512	n=1-bit/peak pixel	n=2- bit/peak pixel	n=3-bit/peak pixel
Lena	14044	28088	42132
Baboon	8082	16164	24246
Forest	18763	37526	56289
Elephant	18833	37666	56499
Water	11666	23332	34998
Tortoise	18724	37448	56172
Average	**15019**	**30037**	**45056**

of block size 256×256 is tabulated. Embedding capacity is significantly increased in comparison to the non block-wise embedding. Further, the cover image is also divided into the block of size 32×32, and results are shown in Table 3.12 which indicates that the embedding capacity has increased in comparison to higher block size. But it was observed that in some cases block of size 32×32 suffers to low peaks which proved to be insufficient to embed all the overhead information in some blocks and fail to recover both the data as well as recovery of the original image. Therefore, it is proposed to take block of size at least

32×32 for error-free data and cover recovery.

3.5.4 Comparative Result Analysis

In this section, comparative analysis of the proposed method along with others state-of-the-art methods has been performed over different performance metrics.

Table 3.13: Data Hiding Capacity Comparison along with Different state-of-the art method

Image	Data Hiding Capacity							
	Proposed Method			Ni	Fallahpour	Garg	Aziz	Kumar
512×512	n=1	n=2	n=3	et al.[177]	et al. [65]	et al.[185]	et al. [186]	et al. [182]
Lena	5353	10706	16059	2899	5126	5126	5126	22908
Baboon	4504	9808	13512	2277	2887	2887	2887	12942
Forest	5569	11138	16707	3652	8118	8118	8118	35999
Elephant	7286	14572	21858	21858	11042	5890	5893	33320
Water	4971	9942	14813	2516	7010	3682	3682	18462
Fire	5294	10588	15882	2537	2712	2712	8118	10293
Tortoise	12754	25508	38262	7067	15150	8694	8694	35382
Village	5431	10862	16293	4976	5231	5240	3919	5100
Human	5707	11414	17121	5111	5301	5313	2985	5136
Average	6318	12637	**18956**	5877	6953	5395	5491	18924

Table 3.13 highlight the data hiding capacity comparison among other [65, 177, 182, 185, 186] state-of-the-art methods. In view of the Table 3.13, our method has the largest data hiding capacity in comparison to the [65, 177, 185, 186], but less data hiding capacity than the Kumar et al. [182]. If the value of n can be taken greater than 3, then our technique may exceed the embedding capacity of the other methods. Results in Table 3.13 demonstrate that the proposed method achieved higher data hiding capacity compared to the other five [65, 177, 182, 185, 186] methods. Table 3.14 highlights the PSNR comparison between the proposed and other five [65, 177, 182, 185, 186] state-of-the-art techniques. PSNR of the proposed technique is less when compared to the other techniques [65, 177, 185, 186] but greater than the technique [182] for the values of $n = 1$ and $n = 2$. However, PSNR does not play a significant role for the proposed method because the receiver receives the

Table 3.14: Peak Signal Noise Ratio comparison along with Different state-of-the art method

Image	Peak Signal Noise Ratio							
	Proposed Method			Ni	Fallahpour	Garg	Aziz	Kumar
512×512	n=1	n=2	n=3	et al.[177]	et al. [65]	et al.[185]	et al. [186]	et al. [182]
Lena	42.30	36.23	31.20	53.68	52.32	52.10	53.69	32.67
Baboon	42.83	37.67	31.40	50.15	51.76	51.61	53.68	35.91
Forest	43.98	37.75	31.50	54.11	51.32	51.05	52.55	33.69
Elephant	43.57	37.11	30.90	52.46	49.33	51.78	53.01	35.81
Water	43.40	36.53	30.53	57.36	49.05	51.56	53.63	31.91
Fire	43.64	37.12	30.78	49.02	54.32	54.78	53.62	32.69
Tortoise	42.85	36.86	30.27	57.36	51.29	51.0	52.52	33.66
Village	42.47	36.45	30.42	42.02	50.03	50.23	53.49	50.22
Human	43.0	37.16	31.20	43.0	49.11	49.76	53.49	49.54
Average	43.11	36.99	30.91	51.01	50.94	**51.54**	53.29	37.54

encrypted marked image and not the marked image alone. Table 3.15 and 3.16 demon-

Table 3.15: Structural Similarity Index comparison along with Different state-of-the art method

Image	Structural Similarity Index							
	Proposed Method			Ni et	Fallahpour	Garg	Aziz	Kumar
512×512	n=1	n=2	n=3	al.[177]	et al. [65]	et al.[185]	et al. [186]	et al. [182]
Lena	0.9994	0.9975	0.9904	0.9998	0.9999	0.9999	0.9999	0.9855
Baboon	0.9993	0.9971	9893	0.9998	0.9999	0.9999	0.9999	0.9863
Forest	0.9997	0.9987	0.9946	0.9999	0.9999	0.9999	0.9999	0.9902
Elephant	0.9994	0.9971	0.9883	0.9998	0.9999	0.9987	0.9998	0.9872
Water	0.9998	0.9976	0.9910	0.9997	0.9999	0.9898	0.9998	0.9841
Fire	0.9994	9978	0.9919	0.9997	0.9998	0.9999	0.9999	0.9867
Tortoise	0.9980	0.9924	0.9696	0.9997	0.9998	0.9998	0.9998	0.9656
Village	0.9988	0.9913	0.9701	0.9976	0.9988	0.9988	0.9992	0.9888
Human	0.9986	0.9956	0.9837	0.9977	0.9987	0.9987	0.9886	0.9788
Average	**0.9988**	0.9936	0.9851	0.9987	0.9987	0.9982	0.9987	0.9836

strate the SSIM metric and POSP metric comparison between the proposed and the other five [65, 177, 182, 185, 186] state-of-the-art techniques, respectively. Both the SSIM value and the POSP value of the proposed versus others' techniques are very close to 1, which shows that the distortion for the marked image is less and tolerable. This shows that the proposed method fulfills both the SSIM metric and the POSP metric properties equally.

Table 3.17 demonstrates the NCC metric between proposed and the other five [65, 177, 182, 185, 186] state-of-the-art techniques, along with nine grayscale images. This metric provides the degree of similarity between the original and marked cover image. Our tech-

nique provides better NCC in comparison to the [65, 177, 182, 185, 186] state-of-the-art techniques.

In view of the above experimental results, our technique achieved better embedding capacity than [65, 177, 182, 185, 186] methods. The experimental result summary is given as follows in turn.

Table 3.16: Proportion of Shifted Pixels comparison along with Different state-of-the art methods

Image	Proportion of Shifted Pixels							
	Proposed Method			Ni et	Fallahpour	Garg	Aziz	Kumar
512×512	n=1	n=2	n=3	al.[177]	et al. [65]	et al.[185]	et al. [186]	et al. [182]
Lena	0.9875	0.9873	0.9873	0.9602	0.9555	0.9534	0.8508	0.9972
Baboon	0.6068	0.9870	0.9842	0.9861	0.9763	0.9758	0.9017	0.9942
Forest	0.9756	0.9749	0.9718	0.9447	0.9458	0.9427	0.6924	0.9965
Elephant	0.9817	0.9814	0.9809	0.9533	0.9727	0.7695	0.3233	0.9944
Water	0.9876	0.9873	0.9872	0.9196	0.9834	0.9690	0.8832	0.9977
Fire	0.9903	0.9901	0.9898	0.9880	0.9562	0.9542	0.6880	0.9982
Tortoise	0.9712	0.9709	0.9657	0.9380	0.9430	0.9356	0.9910	0.9965
Village	0.9886	0.9886	0.9886	9578	0.9555	0.9887	0.8654	0.9877
Human	0.9871	0.9886	0.9871	0.9666	0.9677	0.9873	0.7850	0.9864
Average	0.9418	0.9840	0.9825	0.9571	0.9617	0.9418	**0.7756**	0.9943

Table 3.17: Normalized Cross Correlation comparison along with Different state-of-the art method

Image	Normalized Cross Correlation							
	Proposed Method			Ni et	Fallahpour	Garg	Aziz	Kumar
512×512	n=1	n=2	n=3	al.[177]	et al. [65]	et al.[185]	et al. [186]	et al. [182]
Lena	0.9998	0.9815	0.9965	0.9991	0.99996	0.99996	0.99996	0.9986
Baboon	0.9998	0.99949	0.9980	0.9999	0.99996	0.99996	0.99995	0.9965
Forest	0.9999	0.99974	0.9986	0.9994	0.99997	0.99997	0.99996	0.9964
Elephant	0.9997	0.9988	0.9953	0.9998	0.99996	0.99993	0.99994	0.9956
Water	0.9997	0.99912	0.9966	0.9988	0.99996	0.99992	0.99993	0.9961
Fire	0.9997	0.99877	0.9956	0.9990	0.99998	0.99998	0.99996	0.9982
Tortoise	0.9989	0.99614	0.9836	0.9966	0.99991	0.99991	0.99986	0.9828
Village	0.9997	0.9991	0.9969	0.9988	0.9987	0.9888	0.9999	0.9986
Human	0.9998	0.9991	0.9979	0.9987	0.9988	0.9988	0.9999	0.9856
Average	0.9997	0.99622	0.9949	0.9989	**0.99995**	0.99994	0.99993	0.9949

3.5.5 Experimental Result Summary

This section concludes with the summary of experimental results for all the performance metrics. Average embedding capacity of the proposed method for $n = 1$ is 6.6%, 20.81% and

7.2% greater in comparison to Ni et al. [177], Garg et al.[185] and Aziz et al.[186] methods respectively, whereas Fallahpour et al.[65] 12.91% and Kumar et al. [182] 114.81% methods outperform the proposed technique. Further, it is noticed that the proposed technique for $n = 2$ outperformed Ni et al.[177] by 77.24%, Fallahpour et al.[65] by 60.02%, Garg et al. by 89.02%, and Aziz et al. [186] by 77.80%, whereas Kumar et al.[182] outperformed the proposed method by 54.63%. Moreover, the proposed technique for $n = 3$ outperformed Ni et al.[177] by 104.87%, Fallahpour et al.[65] by 89.97%, Garg et al.[185] by 114.85%, and Aziz et al.[186] by 105.37%, whereas Kumar et al.[182] method outperformed the proposed method by 20.96%. Further, PSNR is also examined in comparison to the other state-of-the-art methods. Our technique has a low PSNR in comparison to the other methods. But this PSNR does not affect the quality of the cover because in our method marked encrypted cover is communicated to the receiver end.

For SSIM, Ni et al.[177] outperformed 0.13%, Fallahpour et al.[65] by 0.14%, and Aziz et al.[186] by 0.14%, whereas the proposed technique outperformed for $n = 1$ Garg et al.[185] by 0.02% and Kumar et al.[182] by 1.49%. For n=2, Ni et al.[177] outperformed the proposed method by 0.61%, Fallahpour et al.[65] by 0.62%, Garg et al. [185] by 0.46%, and Aziz et al. [186] by 0.62%, whereas the proposed method outperformed Kumar et al.[182] by 1.01%. For n=3, Ni et al.[177] outperformed the proposed method by 1.47%, Fallahpour et al.[65] by 1.48%, Garg et al.[185] by 1.32%, and Aziz et al.[186] by 1.48%, whereas the proposed method outperformed Kumar et al.[182] by 0.15%. For $n = 1$, the proposed method outperformed Aziz et al.[186] by 33.39%, whereas Ni et al. [177], Fallahpour et al.[65], and Kumar et al.[182] outperformed the proposed method by 2.88%, 3.51%, and 7.03%, respectively, with Garg et al.[185] showing no significant difference. For $n = 2$, the proposed method outperformed Ni et al. [177] by 2.79%, Fallahpour et al.[65] by 2.15%, the Garg et al.[185] method by 5.66%, and the Aziz et al.[186] method by 38.85%, whereas

the Kumar et al. [182] method outperformed the proposed method by 1.37%. For $n = 3$, the proposed method outperformed Ni et al.[177] by 2.60%, Fallahpour et al.[65] by 1.97%, Garg et al.[185] method by 5.48%, and Aziz et al.[186] by 38.70%, while Kumar et al.[182] method outperformed the proposed method by 1.56%.

Normalized cross correlation for n=1, the proposed method outperformed Ni et al.[177] by 0.0731% and Kumar et al. [182] by 0.4774%, while Fallahpour et al. [65] method outperformed the proposed method by 0.0250%, Garg [185] by 0.0240%, and Aziz et al. [186] by 0.0230%. For n=2, Ni et al. [177] outperformed the proposed method by 0.2757%, Fallahpour et al. [65] by 0.3727%, Garg et al. [185] by 0.3728%, and Aziz et al. [186] method by 0.3718%, while the proposed method outperformed Kumar et al. [182] method by 0.1286%. For n=3, Ni et al. [177] method outperformed the proposed method by 0.4034%, Fallahpour et al. [65] by 0.5014%, Garg et al. [185] by 0.5005%, and Aziz et al.[186] by 0.4995%, whereas the proposed method outperformed Kumar et al.[182] by 0.0010%.

To summarize, the proposed bidirectional histogram shifting and multistage embedding technique shows superior performance against state-of-the-art RDH methods in both qualitative measures and quantitative metrics such as embedding capacity, SSIM, POSP, and NCC on various types of datasets.

3.6 Conclusion

RDH techniques have been extensively investigated and explored for their potential application in different fields. However, most of the RDH techniques hide the overhead bits in the same domain of the cover image, which leads to a reduction in embedding capacity. The proposed multistage RDH approach considered both plain and encrypted domains for secret data communication without any separate channel for overhead data. The proposed

approach exploits histogram peaks for embedding the secret data along with overhead bits, both in plain and encrypted domains. For the experiment, chunks of one, two-, and three-bit secret messages are only used to conceal secret information. Experimental results are performed on different grayscale images and show significant improvement over state-of-the-art approaches. Embedding capacity for smooth and textured grayscale images corresponding to different chunks of bits with their overhead bits is analyzed. on average, the proposed approach achieves embedding capacity 18831 bits for a 64×64 block size of a smooth image, whereas 15962 bits without a block of image size 512×512 for a texture image. In addition, most of the RDH methods perform data embedding in cover either at the plain domain or at the encrypted domain to enhance the secret message embedding capacity, along with minimization of communication overhead. The proposed technique not only achieved optimum embedding capacity but also overcame the issues of communication overhead by exploiting the multistage embedding. Bidirectional embedding of secret data wherein both left peak and right peak from the main histogram peak are considered for secret data 29 embedding, thereby achieving high embedding capacity. In addition, multistage embedding both at the plain domain and encrypted domain not only improved the data security but also overcame the need for separate communication of overhead. Further, it was demonstrated that extending the proposed technique with block-wise embedding of secret data leads to enhanced embedding capacity.

Chapter 4

Design and Development of Encrypted Domain RDH Techniques

4.1 Introduction

Interpolation based Reversible Data Hiding in encrypted images has been explored broadly to achieve high embedding capacity. But achieving the high data hiding capacity along with security is a challenge. Also, communication overhead needs a separate channel, which may lead to the compromise of security. To address this, we proposed a separable reversible data-hiding method, employing bit pair difference at the pixel level. The significance of the proposed work includes an efficient separable RDH technique with high data hiding capacity employing bit-pair differences. Embedding on the bit pairs level not only achieves high secret data embedding capacity but also enhances the security of the secret data along with the cover. The proposed method employs the average and floor function to generate the interpolated cover, which overcomes the overflow/underflow. We proposed a separable method that extracts the secret message or image independently based on available keys rather than in the sequential order, supporting blind data extrac-

tion. The sender's simple rules-based data hiding is considered, and the receiver's data retrieval is achieved by the XOR operation. This overcomes the issues of communication overhead and makes the method computationally inexpensive for its implementation in real time for small/IOT devices. Hence, no separate channel is required for the transmission of overhead information, which also enhances the security of the system. In addition, secure communication of sensitive data over the public network is very critical, especially in the cellular network these days, with the tremendous deployment of the Internet of Things. Generally, this is achieved either by encrypting plain data before transmission or by hiding data under the cover. On the other hand, most of the communication networks are underutilized as they offer high-capacity transmission at multi-gigabit-per-second data rates. Hence, effective utilization of the communication channel along with security and privacy of users is paramount for a channel dedicated to the Internet of Things or Military Applications. Thus, the developed method presents a novel adaptive reversible data hiding method by using cover byte division and multiplication. In this, the cover bytes of the encrypted channel traffic are partitioned into n segments, with the remainder bytes distributed sequentially by incrementing each cover byte by one until the remainder is fully exhausted. In this way, n-copies of encrypted traffic are generated. Further, the secret binary sequence is divided into chunks of bits, and these chunks are converted into the decimal system. Then, these decimal values of the secret message are added to every byte, and a marked encrypted channel traffic is generated. At the receiver, data is retrieved from the remainder, and the cover byte is extracted from the quotient. According to the research done so far, no such method exists that can hide data at the byte level over encrypted traffic without exploiting any correlation between traffic bytes. In addition, existing methods could not achieve high embedding capacity due to less correlation in the encrypted traffic over the channel and the necessity of communication overhead for faithful recovery of

secret data and cover. Ensuring faithful recovery of cover at each user is utmost required to develop the technology which can efficiently utilize channel resources and also ensure privacy and security of multi user data transmission over a common channel. The benefits of this developed method mainly include allowing multiple users to communicate over a common channel without affecting the individual user, to effectively utilize the channel bandwidth and hence the resources, and to enhance the user data security and privacy without any degradation of quality of service. The details of separable employing bit-pair difference at pixel level and adaptive secure data communication over encrypted traffic channel RDH methods are given in the subsequent sections.

4.2 Related Work and Research Gap

Reversible data hiding has been broadly investigated in the last two decades due to its myriad applications in various fields. Generally, RDH work can be classified into lossless compression [4, 30, 183], difference and prediction expansion [8, 28, 146, 183], histogram shifting [9, 34, 65, 182], and interpolation [83, 88, 89, 91]. Details of these work were discussed in the review paper [183].

Interpolation is a promising areathath has been exploited in different directions to achieve high capacity. Interpolation based methods are closely related to the proposed work. Recent evolution in the field of interpolation for RDH is summarized as follows:

Jung and Yoo [15] proposed the first interpolation RDH approach based on interpolation in which first the original cover image of size $2M \times 2N$ was scaled down to size $M \times N$. This scaled-down image was interpolated using Neighbour Mean Interpolation (NMI) by a factor of two, and each block of size 2×2 was interpolated with size 3×3. Non-overlapping blocks of size 2×2 were used for data embedding, in which the absolute value d of the

original and interpolated pixel difference value was calculated. Then, $log_2 d$ bits of secret data were embedded in an interpolated pixel. Embedding capacity varies from 112 - 400 KB with more than 35 dB PSNR. In 2012, Lee and Huang [84] enhanced the embedding capacity of Jung and Yoo [15] by adopting the interpolation by neighbour pixel (INP). In INP, horizontal and vertical interpolated pixel values were determined by taking the average of surrounding two original pixels. However, values of diagonal interpolated pixels were determined by taking the average of the upper and left interpolated pixels. Then, overlapping blocks of size 3×3 were used for data embedding. The maximum value of the original pixel of each interpolated block of size 3×3 was calculated, and then the difference d from this maximum value and the other three interpolated pixels was determined. Lee and Huang [84] achieved a larger embedding capacity than Jung and Yoo [15] due to the maximum value consideration of the original pixel but reduced the quality cover image. In 2013, Chang et al.[83] proposed enhanced neighbour mean interpolation (ENMI) in which horizontal and vertical interpolated pixels were calculated according to the NMI whereas diagonal interpolated pixels were calculated by the average of the neighbouring original four pixels. Further, Chang et al. [83] embedded secret data in two layer. In the first layer, the difference between interpolated and corresponding original pixel was computed, and the secret data was embedded according to the previous method [15], whereas the histogram shifting technique for difference image was adopted for the second layer embedding. This technique achieved a larger embedding capacity in comparison to the Jung and Yoo [15] and better visual quality than NMI method.

In 2017, Malik et al.[187] proposed a modified NMI method in which horizontal and vertical interpolated pixels were determined by the average of two neighbouring pixels and one nearest diagonal pixel, whereas diagonal pixels were computed using the ENMI method. A block of size 2×2 was used for data embedding. For this, the difference of diagonal

interpolated pixels d_1 and the remaining two interpolated pixels d_2 were calculated. The number of data hiding bits was determined by the log function, such as $\log_2 d_1$ and $\log_2 d_2$. Image quality of this method is better than the NMI method, whereas the embedding capacity outperformed the Jung and Yoo [15] method. In 2017, Zhang et al.[90] proposed a parabolic interpolation (PI) method. Then, interpolated pixels were calculated using the parabolic equation $y = ax^2 + xb + c$ in horizontal, vertical, and diagonal directions. For determining the interpolated value, x_i values were considered as the index of the pixel, whereas y_i represents the pixel values. With the help of this information, first the value of constant coefficients a, b, c were determined, and then interpolated pixels y_i values were calculated. A block of size 5×5 was chosen for data embedding and the average of the two neighbouring original pixels was determined, and then the difference d between the interpolated pixel and average value was determined. The large value of d leads to the high distortion in the cover image. To minimize this problem, a threshold T was defined to control the parameter by deciding the embeddable interpolated pixels. Embedding capacity and PSNR of PI methods are better than Jung and Yoo [15] and Lee and Hung [84] methods.

Xiao et al. [188] proposed interpolation based separable RDH-EI. In this, the content owner first estimated the location for data embedding using interpolation and generated the location map and then encrypted the original cover image. Most valid bits of the encrypted cover were flipped, corresponding to the location map to embed the secret message. Further, Malik et al. [189] introduced a Paillier cryptosystem based interpolation technique for RDH-EI. In this, an interpolation technique was used to estimate the most significant pixel bit to create the location map and check if the pixel is suitable for embedding. Then, the location map was compressed using the Paillier cryptosystem, and the image was encrypted. At the receiver side, firstly cover image is decrypted and then, the extracted secret data is encrypted to recover the original image. In 2023, Punia et al. [190] pro-

posed interpolation based RDH wherein interpolated pixels were calculated by taking the average of the square root of the product of two original neighboring pixels and another neighboring pixel. The average of two interpolated pixels was taken for the central pixel. For data embedding, the first RSA algorithm was applied on the secret message, and then data embedding was performed based on the range of group intensity. Further, in 2024, Xiong et al. [191] introduced a Logistic map based interpolation RDH method for encrypted images to enhance the complexity. For this, bit-rearrangement and diffusion function were adopted for cover encryption. The NMI method was applied to interpolate the encrypted cover. Congruence modulo operation was used to embed the secret message.

In sum, interpolation based RDH was investigated broadly to obtain high data hiding capacity without much more distortion in image quality. But, still obtaining the high capacity along with security is a challenge. Also, communication overhead needs separate channels, which may lead to a compromise in security. To address this, we proposed a Separable RDH Employing Bit Pair Differences at the pixel level wherein no overhead information is required during the communication. The proposed method employ data hiding at the bit-pair level, which embeds more message bits per pixel in comparison to the existing RDH methods. It is very difficult to identify the message bits in a marked pixel because the attacker does not know which bit-pair is used to hide the message bit and in which order it is considered. Additionally, the proposed method also embeds the encrypted message. In view of these points, the proposed method enhances the embedding capacity and security and avoids the overflow/underflow problem due to the embedding at the bit-pair level. The core design of the proposed work is presented in the next section.

4.3 Proposed Framework for Secret Message Communication of employing Bit-Pair Differences

This section proposes the separable interpolation based RDH method using bit pair difference at the pixel level. The general framework of the proposed method is depicted in Fig. 4.1. In this, the secret message embedding method consists of data pre-processing and data embedding stages. In stage (a), plain message and cover image $\mathcal{I}$ of size $\mathcal{M} \times \mathcal{N}$ is pre-processed before the message is embedded. Firstly, the plain message is formulated into blocks of 16 bits. Encryption engine converts these message blocks into a secure secret message using key $\mathcal{K}_1$ to enhance the message security, which is to be embedded into the interpolated cover. Further, the cover image is encrypted through the encryption engine using key $\mathcal{K}_2$ to enhance the cover security. Then, the encrypted cover is interpolated using the floor and average function to embed the secure secret message.

In the data hiding stage, the data embedding method is employed at the bit-pair level on interpolated pixels and generates the marked encrypted cover. This marked encrypted cover is sent to the receiver, wherein the receiver first segregates the interpolated and encrypted cover pixels. The data extraction method is applied to interpolated pixels at the bit-pair level. This extracts the secure secret message, and further, this message is decrypted through the decryption engine using key $\mathcal{K}_1$ to generate the plain message. For cover recovery, interpolated rows and columns are discarded from the marked encrypted cover. Then, the encrypted cover is decrypted through the decryption engine using key $\mathcal{K}_2$ to generate the cover image. The detailed explanation of the core design of data embedding is given in Section 4.3.1, followed by the data extraction phase in Section 4.3.2.

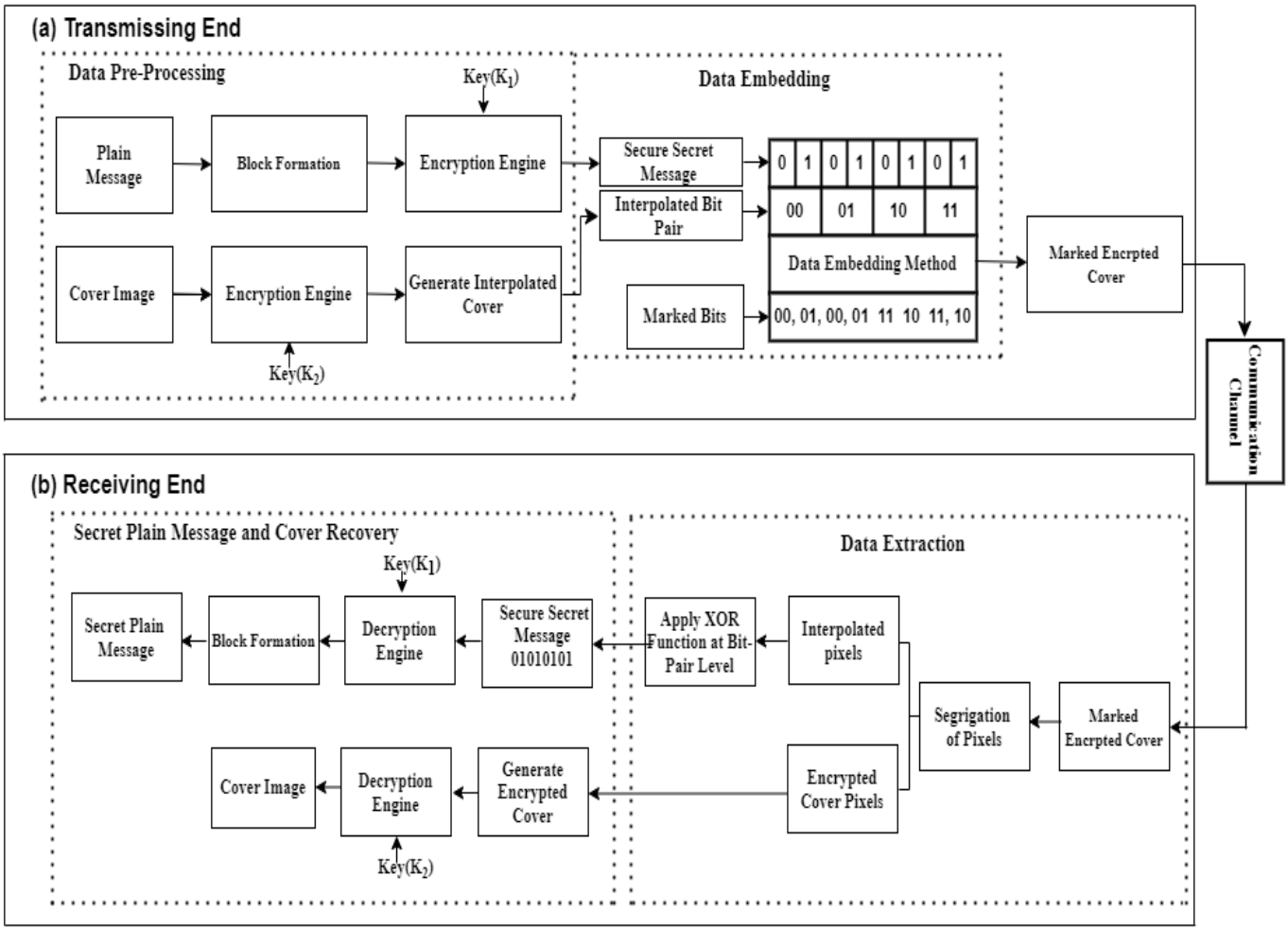

Fig. 4.1 General framework of Proposed Method: a) Transmission end: cover data is first encrypted and secret message is embedded. b) Receiver End: Secret message is recovered with very simple $\mathcal{XOR}$ operation and cover is obtained without any overhead requirement.

4.3.1 Core Design of Secret Data Embedding Method

The proposed data embedding method consists of two stages, mainly pre-processing and data embedding. The overview of the proposed secret data embedding method is depicted in Fig. 4.2, wherein red colour shows the interpolated and others are non-interpolated pixels. Algorithm 11 illustrates the pseudo code for the proposed data embedding method. The details of the secret message embedding method are as follows.

The secret data embedding phase consists of pre-processing and data embedding stages. In stage(a), the cover image $\mathcal{I}$ of size $\mathcal{M} \times \mathcal{N}$ is pre-processed before the secure message embedding. Firstly, the cover is encrypted using Advanced Encryption Standard ($\mathcal{AES}$) to

enhance the cover security. Mathematically, it is defined by the Eq 4.3.1.

$$e(\mathcal{I}) = \mathcal{E}(\mathcal{K}, \mathcal{P}_i)_{i=1}^{\mathcal{M}\times\mathcal{N}} \tag{4.3.1}$$

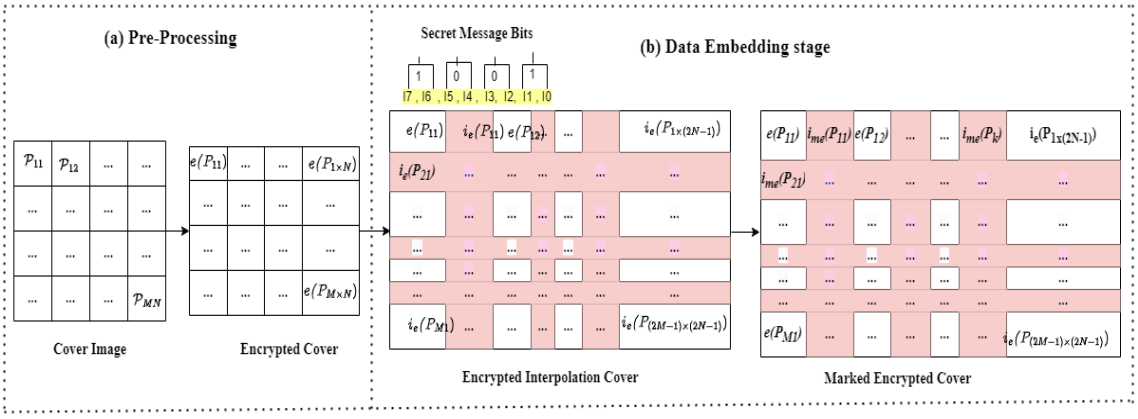

Fig. 4.2 Proposed Data Embedding Method. a) Pre-processing stage convert the plain cover into encrypted cover using $\mathcal{AES}$ Encryptor. (b) Data embedding stage generate rooms for secret message through interpolation followed by secret message embedding at pixel level.

where $\mathcal{E}$ is $\mathcal{AES}$ based encryption function, $\mathcal{K}$ is secret encryption key and $\mathcal{P}_i$ is i^{th} pixel of cover($\mathcal{I}$). This encrypted cover is further interpolated employing average and floor functions to generate interpolated image $\mathcal{I}_p$ of size $(2\mathcal{M}-1)\times(2\mathcal{N}-1)$. For this, an empty row and column are added between two adjacent rows and columns. The values of empty rows and columns are calculated using the average interpolation function by taking the average of two adjacent pixels. Overflow is controlled by using the floor function of the average of two adjacent pixels. Mathematically, an interpolated image can be generated by the Eq 4.3.2.

$$\mathcal{I}_p = (\lfloor \frac{e(\mathcal{P}_i) + e(\mathcal{P}_{i+1})}{2} \rfloor)_{i=1}^{(2\mathcal{M}-1)\times(2\mathcal{N}-1)} \tag{4.3.2}$$

where, $e(\mathcal{P})_i$ denotes the i^{th} encrypted pixel. Interpolated encrypted cover $\mathcal{I}_p$ is generated, and this cover is further used to hide the secret message.

In stage 2, interpolated pixels are used to embed the secret message, and other pixels are unchanged during the secure secret message embedding. For this, each i^{th} interpolated

pixel of encrypted image is converted into the binary eight-bit binary sequence, which is defined as in Eq 4.3.3.

$$e(\mathcal{P}_i)_2 = l_{i7}, l_{i6}, l_{i5}, l_{i4}, l_{i3}, l_{i2}, l_{i1}, l_{i0} \tag{4.3.3}$$

where, l_i denotes the lsb-bit of i^{th} pixel. In this, bits pair $l_{i7}l_{i6}, l_{i5}l_{i4}, l_{i3}l_{i2}$ and $l_{i1}l_{i0}$ are separated to perform the secret message embedding process at bit-pair level. This process generates the interpolated encrypted marked cover $(\mathcal{I}_{me})$. Every bit-pair can be either $00, 01, 10$, or 11. For each bit pair corresponding secret message bit can be either 0 or 1 accordingly. Mathematically, the marked bit-pair is generated by the function $f(m, b_p)$ as in Eq 4.3.4.

$$b'_p = f(m, b_p) \tag{4.3.4}$$

where, m, b_p, and b'_p denote the secret message bit, cover bit-pair, and marked bit-pair, respectively. The generation of the marked encrypted image is determined under the following eight cases.

Case 1: If the bit pair is 00 and the corresponding secret message bit is zero, then no change is carried out in the bit pair. On the other hand, if the secret message bit is 1, then the bit pair 00 changes to 01 bit pair. For this case, a mathematically marked bit-pair is generated by the Eq 4.3.5.

$$f(i_{b1}, i_{b0}, m) = \begin{cases} 00 & \text{if } m = 1, i_{b1} = i_{b0} = 0 \\ 01 & \text{if } m = i_{b1} = i_{b0} = 0 \end{cases} \tag{4.3.5}$$

Case 2: If bit pair is 01 and corresponding secret message bit is 0, then bit pair 01 changes to 00 bits, whereas bit pair remains the same, if secret message bit is 1. For this case,

a mathematically marked bit-pair is generated by the Eq 4.3.6.

$$f(i_{b1}, i_{b0}, m) = \begin{cases} 01 & \text{if } m = 0, i_{b1} = 0, i_{b0} = 1 \\ 00 & \text{if } m = 1, i_{b1} = 0, i_{b0} = 1 \end{cases} \tag{4.3.6}$$

Case 3: If bit pair is 10 and corresponding secret message bit is 0 then bit pair 10 changes to 11 bit pair whereas if secret message bit is 1 then bit pair 10 is unchanged. For this case, a mathematically marked bit-pair is generated by the Eq 4.3.7.

$$f(i_{b1}, i_{b0}, m) = \begin{cases} 11 & \text{if } m = 0, i_{b1} = 1, i_{b0} = 0 \\ 10 & \text{if } m = 1, i_{b1} = 1, i_{b0} = 0 \end{cases} \tag{4.3.7}$$

Case 4: If the bit pair is 11 and the corresponding secret message bit is zero, then no change is carried out in the bit pair, whereas if the secret message bit is 1, then bit pair 11 changes to 10 bit pair. For this case, a mathematically marked bit-pair is generated by the Eq 4.3.8.

$$f(i_{b1}, i_{b0}, m) = \begin{cases} 11 & \text{if } m = 0, i_{b1} = i_{b0} = 1 \\ 10 & \text{if } m = 1, i_{b1} = i_{b0} = 1 \end{cases} \tag{4.3.8}$$

On the basis of the above four cases, a marked encrypted cover is generated. These cases are also summarized in Table 4.1. In this, the bit-pair is changed in such a way that after applying the $\mathcal{XOR}$ function on the resultant bit-pair, the secret message bit is produced. Here, the interpolated bit pair is replaced by the resultant bit pair to generate the marked encrypted interpolated image. The marked encrypted image is forwarded to the receiver end, where the secret message and cover are extracted.

In sum, the proposed data embedding method consisted of two stages, mainly pre-

Table 4.1: Summary of Proposed Data Embedding Method

Cover Bit Pair	00		01		10		11	
Secret Message Bit	0	1	0	1	0	1	0	1
Action	No Change	Add 1	Subtract 1	No Change	Add 1	No Change	No Change	Subtract 1
Resultant Bit Pair	00	01	00	01	11	10	11	10

processing and data embedding. Data embedding at the bit-pair level not only achieves high secret message embedding but also enhances the security of the secret message with the cover. Due to the embedding at bit-pair level, the overflow/underflow issue is also resolved. A simple rule is applied for secret message embedding. For this, the embedding process is computationally inexpensive and can be implemented for real-time applications. In the next subsection, details of the data extraction method are presented.

4.3.2 Core Design of Data Extraction Technique

The data extraction method is designed to extract the secret message along with the original cover. The data extraction phase consists of three stages: secret message extraction, pre-processing of the encrypted image after secret message extraction, and recovery of the original cover. Core design architecture of data extraction and plain cover recovery phase is depicted as in Fig. 4.3 . Pseudo code of proposed data extraction and cover recovery method is shown is in algorithm 12.

The receiver receives the marked interpolated cover $(\mathcal{I}_{me})$, and this image is processed for data extraction. Secret message bits are extracted from each encrypted interpolated pixel.For this, each encrypted marked interpolated pixel is converted into an eight-bit binary sequence and it is defined by the Eq 4.3.9.

$$\mathcal{I}_{me}(\mathcal{P}_i) = b'_{i7}, b'_{i6}, b'_{i5}, b'_{i4}, b'_{i3}, b'_{i2}, b'_{i1}, b'_{i0} \tag{4.3.9}$$

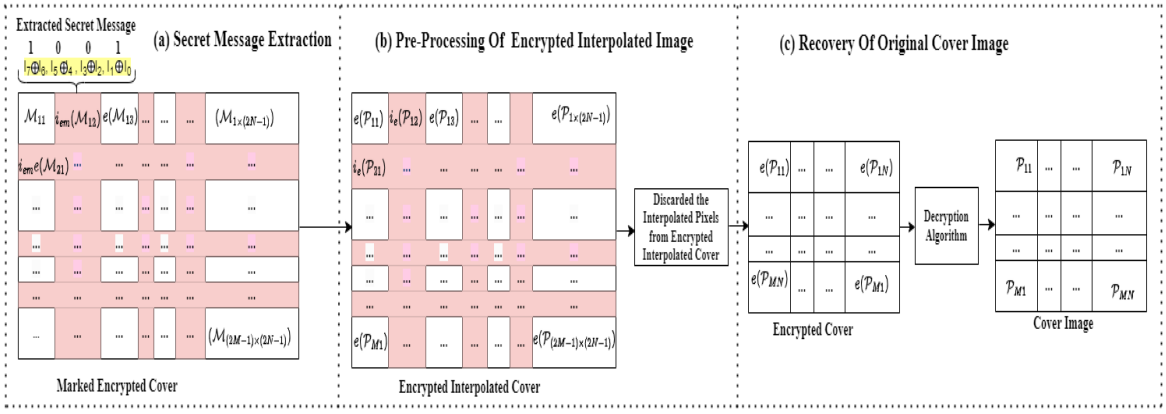

Fig. 4.3 Proposed Data Extraction and Cover Recovery Method. (a) Secret Message Extraction. (b) Pre-processing stage converts the encrypted interpolated cover by discarding the interpolated rows and columns. (c) Recovery of original cover by decryption the encrypted cover using AES Decryption key.

where,$\mathcal{I}_{me}(\mathcal{P}_i)$ denotes the i^{th} pixel of marked encrypted cover. Further, bit-pairs are separated for the extraction of the secret message bit. Then, $\mathcal{XOR}$ function is performed on individual bit-pairs, and the result of this function gives the message bit. Secret Message bits $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4$ are extracted from each bit-pair. Equations from 4.3.10 to 4.3.13 extract these message bits from different bit pairs as follows.

$$\mathcal{M}_1 = b'_{i1} \oplus b'_{i0} \tag{4.3.10}$$

$$\mathcal{M}_2 = b'_{i3} \oplus b'_{i2} \tag{4.3.11}$$

$$\mathcal{M}_3 = b'_{i5} \oplus b'_{i4} \tag{4.3.12}$$

$$\mathcal{M}_4 = b'_{i7} \oplus b'_{i6} \tag{4.3.13}$$

Secret message bits are extracted from each interpolated pixel and then appended with the secret message bits to get the full length of the secret message. The encrypted interpolated image is pre-processed after message extraction. For this, each interpolated row and column is discarded from the encrypted interpolated cover. Then, the encrypted cover is further decrypted using $\mathcal{AES}$ decryption algorithm and the original cover is recovered.

It can be defined by the Eq 4.3.14.

$$\mathcal{P}_i = \mathcal{D}(\mathcal{K}, \mathcal{E}) \tag{4.3.14}$$

where $\mathcal{D}$ is $\mathcal{AES}$ based decryption and $\mathcal{K}$ is secret key used.

The proposed method supports blind extraction wherein secret message extraction and cover image recovery process can be applied independently without any specific order for recovery. In the proposed method, data embedding is performed in the encrypted interpolated cover, thereby noise in the cover image or slight variation in the cover image does not affect the data embedding process. At the receiver side, the receiver extracts the secret message correctly and recovers the noisy cover image. On the other hand, channel related noise will be handled separately by the employed channel error correction methods. Hence, the proposed method is robust to noise introduced, or in other words, even though there are slight variations in the cover image, the hidden secret message is recovered faithfully.

In sum, the proposed data extraction method consists of three stages: mainly secret message extraction, pre-processing of encrypted interpolated cover, and recovery of the original cover. This method is based on a simple rule; it is computationally inexpensive, and no overhead communication is required. Since it's computationally inexpensive, this method can be implemented in real time and used in fields such as defence, medical, law enforcement etc., in real time. Further, the proposed method is experimentally validated, and details are provided in the next section. The pseudo code of data extraction is given in Algorithm 12 as follows.

4.4 Experimental Validations

The performance of the proposed method is evaluated over eight state-of-the-art RDH methods. Experimental setup and performance metrics used are described in section 4.4.1. The qualitative comparison analysis is discussed in section 4.4.2, followed by the quantitative comparison analysis with existing RDH methods in section 4.4.3.

4.4.1 Experimental Design

We have implemented the existing state-of-the-art methods of Lee and Haung[84], Jung and Yoo [15], Chang et al. [192], Zhang et al. [90], Malik et al. [193], Fatuma and Adnan [194], Zhong et al. [195], Bai et al.[196] with the proposed method on the python programming platform. The hardware platform used is a 1.6 GHz Dual-Core Intel Core i5 with 8 GB 2133 MHz LPDDR3 memory. The next subsection contains a description of the qualitative results analysis.

4.4.2 Qualitative Results

Eight grayscale test images, i.e., Baboon, Boat, Airplane, Bridge, Cameraman, GoldHill, House, Peppers of size 512×512 are considered for the experiment. The test images are shown in Fig. 4.4. The set of secret data for embedding is generated randomly. Three metrics are used to evaluate the performance of the proposed method, namely Embedding capacity in (bits), Peak Signal Noise Ratio (PSNR) in (dB), and Bit Rate in (bpp).

The visual quality of the stego-image is measured by the PSNR metric. For an image $\mathcal{I}$ of size $\mathcal{M} \times \mathcal{N}$ and its distorted version $\mathcal{X}$, the PSNR of image $\mathcal{X}$ concerning the original

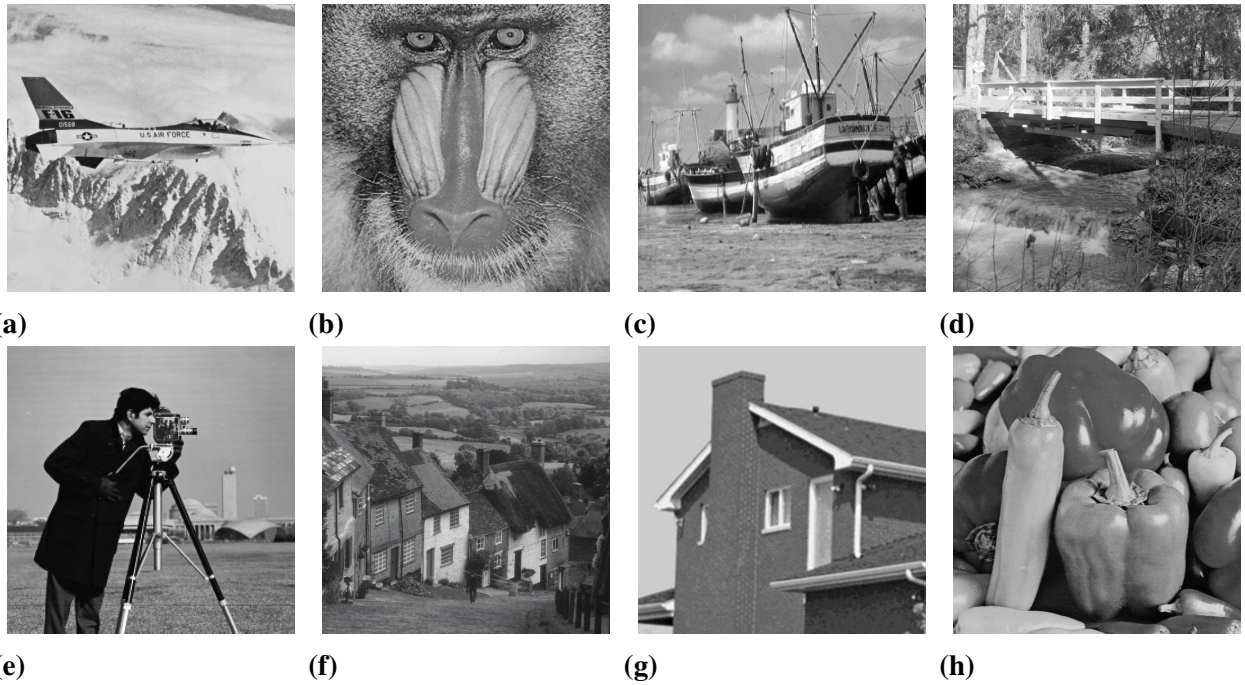

Fig. 4.4 Test Images (a)JetPlane, (b)Baboon, (c)Boat, (d) Bridge, (e) Cameraman, (f)GoldHill, (g)House, (h)Peppers

cover image $\mathcal{I}$ can be calculated using the following Eq.4.4.1.

$$PSNR = 10 \times log_{10}(\frac{255^2}{MSE}) \tag{4.4.1}$$

where, MSE represents the mean square error, which can be defined by the Eq 4.4.2.

$$MSE = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(\mathcal{I}(i,j) - \mathcal{X}(i,j))^2}{\mathcal{MN}} \tag{4.4.2}$$

The cover $\mathcal{I}(i,j)$ and stego-image $\mathcal{X}(i,j)$ are the intensity values of the pixels in the i^{th} row and the j^{th} column.PSNR value is proportional to visual quality, as PSNR value increases then visual quality is also improved.

Embedding capacity measures the total amount of bits that can be embedded into the image by adopting a data hiding scheme. This can be defined mathematically as in Eq. 4.4.3.

$$Total\ Embedding\ Capacity = Image\ size \times Bit\ Rate \tag{4.4.3}$$

Bit-rate measures the number of bits of message that can be embedded into each pixel of an image. The qualitative analysis with respect to the embedding capacity and visual quality is given as follows.

a) **Embedding Capacity Comparison**: The proposed method has the highest embedding capacity in comparison to Jung and Yoo [15], Chang et al.[192], Zhong et al. [195], Bai et al. [196] except for Lee and Haung's [84] method, which is shown in the Fig. 4.5. However, Malik et al. [193] have shown better embedding capacity for Airplane, Bridge, and House test images, whereas method [194] has higher embedding capacity for all eight test images. It is also shown in Fig. 4.5 that Embedding capacity is highest for 3 and 4 bpp in comparison to the [84, 90, 192, 193, 194, 195, 196] methods for all the test images.

b) **Visual Quality Comparison**: The proposed method at bit rates of 3 and 4 bpp achieved the highest visual quality in comparison to [84, 90, 193, 194, 196] methods for eight images. However, Chang et al. [192] achieved higher visual quality for Boat, Airplane, goldHill, and Peppers images, and Zhong et al. [195] achieved higher visual quality for all eight test images. The proposed method employs secret message embedding in the interpolated encrypted cover. For the interpolated encrypted cover, maintaining visual quality does not play any pivotal role in gauging the performance of the method. Hence, the trade-off between embedding capacity and visual quality should not be considered as a significant parameter for deciding the application of the proposed method, which can be effectively utilized for encrypted domain applications.

The proposed method supports blind extraction wherein secret message extraction and cover image recovery process can be applied independently without any specific order for recovery. In the proposed method, data embedding is performed in the encrypted interpolated cover, thereby noise in the cover image or slight variation in the cover image does not affect the data embedding process. At the receiver side, the receiver extracts the secret

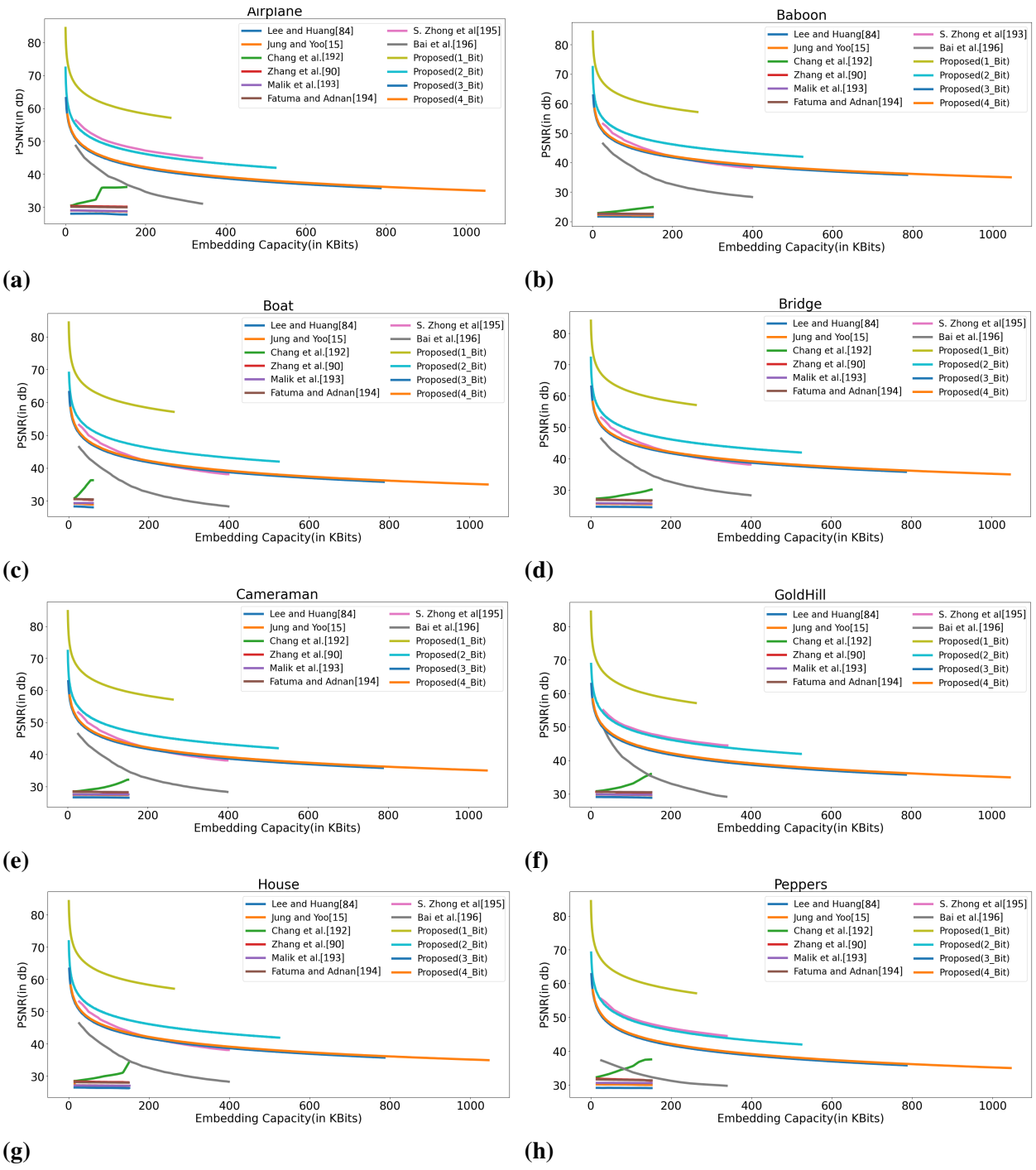

Fig. 4.5 Test Images (a)JetPlane, (b)Baboon, (c)Bridge, (d)Boat, (e)Peppers, (f)GoldHill, (g)Cameraman, (h)House

message correctly and recovers the noisy cover image. On the other hand, channel related noise will be handled separately by the employed channel error correction methods. Hence, the proposed method is robust to noise introduced or there are slight variations in the cover image, and hence the hidden secret message is recovered faithfully. Qualitative comparison is shown in Fig. 4.5 as a graphical representation. Quantitative analysis of the

proposed method is presented and discussed as follows in turn.

4.4.3 Quantitative Results

We have evaluated the embedding capacity, visual quality, and bit rate per pixel of the proposed method and performed quantitative comparison analysis with the eight existing RDH methods in this section. Table 4.2 demonstrates comparison results of the proposed method with other state-of-the-art methods based on PSNR(dB), Embedding capacity(in bits), and Bit Rate/Embedding rate(in bpp) obtained on the eight test images.

The proposed method shows the best embedding capacity when compared to the other eight state-of-the-art schemes [15, 84, 90, 192, 193, 194, 195, 196] on eight test images. The embedding capacity of the proposed method ranges between 2,62,000(PSNR = 57.15 dB and bit rate = 1 bpp) and 10,45,000(PSNR = 35 dB and bit rate = 4 bpp) bits that are much higher than the other existing methods which have the maximum embedding rate of 7,52,473(PSNR = 25.96 dB and bit rate = 2.87 bpp) bits. It is observed from Table 4.2 that proposed method at bit rate of 2 bpp has higher embedding capacity than Jung and Yoo [15], Chang et al.[192], Zhong et al. [195], Bai et al. [196] except for Lee and Haung's [84], which has 1,16,938 more embedding capacity for Baboon test image at bit rate of 1.75 bpp, Malik et al [193] which has better embedding capacity for Airplane, Bridge and House. Fatuma and Adnan [194] have the embedding capacity higher for all eight test images. The proposed method, on the other hand, achieves the best embedding capacity among the other eight existing methods on all test images at bit rates of 3 and 4 bpp.

Proposed method at bit rate of 3 and 4 bpp, achieved the highest PSNR in comparison to the other eight existing methods in all the test images except for Chang et al. [192] and S. Zhong et al. [195]. Chang et al. [192] achieved higher PSNR for the test images of Boat(PSNR = 36.88 dB, bit rate = 0.83 bpp), Airplane(PSNR = 36.19 dB, bit rate = 0.70

bpp), GoldHill(PSNR = 37.13 dB, bit rate = 0.91 bpp), and Peppers(PSNR = 38.05 dB, bit rate = 0.81 bpp). Zhong et al. [195] achieved higher PSNR values for all eight test images, i.e., between 39.61 dB and 45 dB with bit rate in the range of 1.3 to 1.5 bpp.

The proposed method successfully generates high quality marked images even at high embedding rates of 3 and 4 bpp. It shows an average PSNR of 57.15 dB for all the test images at an embedding rate of 1 bpp, 41.97 dB at 2 bpp, 37.79 dB at 3 bpp, and 35 dB at 4 bpp, which is much higher than the other methods which achieve the maximum PSNR of 45 dB at 1.3 bpp. Quantitative analysis shows that the proposed method achieved an average higher embedding capacity with high visual quality of the stego-image. In the proposed method, redundancy among the pixels was not considered as the embedding is performed on the encrypted interpolated cover, thereby any characteristic of pixels or image does not affect the performance of the proposed method. Hence, the proposed method is independent of the cover image and performs equally well on grayscale and color images. In addition, the method's performance does not vary with the complexity or texture of the image. The proposed method is based on a customized bit rate property that can be configured according to the needs of the user between 1 to 4 bpp. This further shows the superiority of the proposed algorithm that presents the capability to have a high embedding rate of $\leq$ 4 bpp with knowledgeable PSNR, while the other compared methods have a maximum embedding rate of $\leq$ 2.87 bpp. The proposed method is not only very simple to implement but also has no limitations to images with specific patterns or characteristics. Proposed separable reversible data hiding method employing bit pair shows superior performance against state-of-the-art RDH methods for embedding capacity and PSNR on different types of cover. Also, embedding capacity is independent of the cover type.

Algorithm 11 Pseudo Code of Proposed Data Embedding Method

Input: Interpolated Image
Output: Marked Interpolated Image
message = np.array([])
def $change\_bit(lsb_1, lsb_0, message\_bit)$:
if then $(lsb_1 == 0) and (lsb_0 == 0) and (message\_bit == 0)$:
    return 0
else if then $(lsb_1 == 0) and (lsb_0 == 0) and (message\_bit == 1)$:
    return +1
else if then $(lsb_1 == 1) and (lsb_0 == 0) and (message\_bit == 0)$:
    return +1
else if then $(lsb_1 == 1) and (lsb_0 == 0) and (message\_bit == 1)$:
    return 0
else if then $(lsb_1 == 0) and (lsb_0 == 1) and (message\_bit == 0)$:
    return -1
else if then $(lsb_1 == 0) and (lsb_0 == 1) and (message\_bit == 1)$:
    return 0
else if then $(lsb_1 == 1) and (lsb_0 == 1) and (message\_bit == 0)$:
    return 0
else if then $(lsb_1 == 1) and (lsb_0 == 1) and (message\_bit == 1)$:
end if
return -1
def insert_data(input_image):
global message
row_count = input_image.shape[0]
col_count = input_image.shape[1]
image = np.copy(input_image)
old_row_count = (row_count+1)//2
old_col_count = (col_count+1)//2
$size = (row\_count * col_count) - (old\_row\_count *$
old_col_count)
random_array = np.random.randint(2, size=size)
message = np.append(message, random_array)
k = 0
for i **do** in range(0, row_count, 2):
    for j **do** in range(1, col_count, 2): message_bit= random_array[k] interpolated_value = image[i][j]
$lsb_0 = interpolated\_value \& 1$
$lsb_1 = (interpolated\_value >> 1) \& 1$
$image[i][j] += (change\_bit(lsb_1, lsb_0, message\_bit))$
        k += 1
    end for
end for
for i **do** in range(1, row_count, 2):
    for j **do** in range(0, col_count): message_bit = random_array[k] interpolated_value = image[i][j] $lsb_0 = interpolated\_value \& 1$ $lsb_1 = (interpolated\_value >> 1) \& 1$ image[i][j] += $change\_bit(lsb_1, lsb_0, message\_bit)$ k += 1
    end for
end for
return Marked Interpolated Image

Algorithm 12 Pseudo Code for Data Extraction Method

```
Input Marked Cover Image
Output Secret Message
input_image= np.array(Image.open("images/girl.png"))[:,:,:]
row = input_image.shape[0]
col = input_image.shape[1]
inserted_image_1 , inserted_image_2 = sender(input_image)
recovered_image, Received_message =receiver(inserted_ image_1 , inserted_image_2)
error = 0
for i in range(0,row) : do
    for j in range(0,col) : do
        for k in range(0,3) : do
            if (recovered_image[i][j][k]! = input_image[i][j][k]) : then
                error+ = 1
            end if
        end for
    end for
end for
correct=1
for k in range(0,len(message)): do
    if (message[k]!=Received_message[k]): then
        correct=0
        break
    end if
end for
if (correct==1) then:
    Received Message is Correct
end if
```

Table 4.2: Comparison Table along with Eight State-of-the-art Interpolation based Methods

Methods	**Metrics**	**Images for Experiment**							
		Baboon	**Boat**	**Airplane**	**Bridge**	**Cameramen**	**Goldhill**	**House**	**Peppers**
Lee and Hung [84]	Capacity	640938	384669	342520	516058	488169	443245	411072	38881
	Bit Rate	1.75	1.46	1.30	1.96	1.86	1.69	1.56	1.48
	PSNR	21.20	27.41	27.38	23.75	25.66	28.35	26.09	28.66
Jung and Yoo [15]	Capacity	428240	216258	177830	332834	285449	251003	244607	197605
	Bit Rate	1.63	0.82	0.67	1.26	1.08	0.95	0.93	0.75
	PSNR	21.93	28.73	28.60	25.13	26.94	29.44	27.06	29.91
Chang et al [192]	Capacity	382841	219442	184280	267138	261441	240232	225602	213837
	Bit Rate	1.46	0.83	0.70	1.01	0.99	0.91	0.86	0.81
	PSNR	29.32	36.88	36.19	33.41	34.91	37.13	34.68	38.05
Zhang et al [90]	Capacity	614981	551986	528793	589130	572492	565988	558878	531550
	Bit Rate	2.30	2.10	2.01	2.24	2.18	2.15	2.13	2.02
	PSNR	22.14	728.85	28.70	25.64	27.08	29.11	26.99	30.08
Malik et al [193]	Capacity	401420	416718	632874	575020	495016	431024	532830	446328
	Bit Rate	1.53	1.58	2.41	2.19	1.88	1.64	2.03	1.70
	PSNR	22.41	29.15	28.10	25.41	27.23	29.67	26.78	30.26
Fatuma and Adnan [194]	Capacity	699923	541744	726450	752473	647365	571052	679718	537330
	Bit Rate	2.66	2.06	2.77	2.87	2.46	2.17	2.59	2.04
	PSNR	22.32	29.67	28.91	25.96	27.29	29.98	27.43	30.49
S. Zhong et al [195]	Capacity	393216	393216	340787	393216	393216	340787	393216	340787
	Bit Rate	1.50	1.50	1.30	1.50	1.50	1.30	1.50	1.30
	PSNR	39.61	39.61	45	39.61	39.61	44.63	39.61	43.90
Bai et al [196]	Capacity	393216	393216	340787	393216	393216	340787	393216	340787
	Bit Rate	1.50	1.50	1.30	1.50	1.50	1.30	1.50	1.30
	PSNR	29.86	29.86	31.11	29.86	29.86	29.05	29.86	29.01
Proposed method(1_*bit*)	Capacity	262000	262000	262000	262000	262000	262000	262000	262000
	Bit Rate	1	1	1	1	1	1	1	1
	PSNR	57.15	57.13	57.15	57.15	57.16	57.15	57.14	57.15
Proposed method(2_*bit*)	Capacity	524000	524000	524000	524000	524000	524000	524000	524000
	Bit Rate	2	2	2	2	2	2	2	2
	PSNR	41.96	41.97	41.96	41.98	41.96	41.97	41.97	41.97
Proposed method(3_*bit*)	Capacity	786000	786000	786000	786000	786000	786000	786000	786000
	Bit Rate	3	3	3	3	3	3	3	3
	PSNR	35.802	35.790	35.785	35.793	35.793	35.788	35.784	35.789
Proposed method(4_*bit*)	Capacity	1045000	1045000	1045000	1045000	1045000	1045000	1045000	1045000
	Bit Rate	4	4	4	4	4	4	4	4
	PSNR	34.99	35.00	34.99	34.99	35.00	34.99	34.99	35.00

4.5 Adaptive Secure Data Communication over Encrypted Channel Traffic

Data hiding techniques have undergone significant transformations over the past, evolving from steganography to reversible data hiding over the plain or encrypted traffic. In this, the recovery of exact cover remains a challenge, considering the trade-off between capacity and security. In this direction, Reversible Data Hiding (RDH) has been applied significantly for copyright protection, source tracing, annotation of photographs, and many more. Various techniques of RDH have been introduced for an unencrypted communication channel. For instance, RDH approaches can be broadly classified into Histogram Shifting, Difference Expansion, Prediction Error Expansion, Pixel Value Ordering, and Most Significant Bits. However, these techniques are generally applied to images, video, audio, and other types of cover with block-level segregation of data. In addition, these techniques require a large volume of overhead bytes that need to be communicated to the receiving end for recovering the data and the cover faithfully. This not only reduces data hiding capacity but also imposes security concerns.

Secure communication of sensitive data over the public network is very critical, especially in the private/public 5G/6G cellular network with tremendous deployment of the Internet of Things. Generally, this is achieved either by encrypting plain data before transmission or by hiding data under the cover. On the other hand, most of the communication networks, especially 5G/6G cellular networks, are underutilized as they offer high-capacity transmission at multi-gigabit-per-second data rates. Hence, effective utilization of the communication channel along with security and privacy of users is paramount for a channel dedicated to the Internet of Things or Military Applications. Recently, some researchers

explored the dual image (DI) RDH technique to improve the data embedding capacity and visual quality of the cover image.

The proposed method presents a novel adaptive reversible data hiding system using a byte division and multiplication approach. In this, the method hides the user message adaptively as the number of bits of the secret user message is embedded on each byte of cover data, generating the marked encrypted channel traffic which is extracted from it through the channel prerequisite or other pre-determined parameters. The major objectives fulfilled by our method are as follows:

- To hide the user's secret data over an encrypted traffic at the byte level without any communication overhead.
- To adaptively embed secret data to obtain high reversible data hiding capacity.
- To allow multiple users to communicate over a common channel without affecting the individual user
- To effectively utilize the channel bandwidth and hence the resources.
- To enhance the user data security and privacy without any degradation of the quality of service.

4.5.1 Proposed Adaptive Secure Communication System

The proposed system discusses a secure data communication system for transmitting a user's secret message over an encrypted channel traffic and the extraction of the secret message and the encrypted channel traffic in detail. An overview of the proposed method is depicted in Fig. 4.6. The secure data communication system includes a data hiding system that hides the secret message in the encrypted traffic at the byte level to generate

marked encrypted traffic. This encrypted traffic is transmitted over the public/private network and communicated to the data extraction system. The method processes a single traffic byte and is used to embed the secret data. The method is independent of the type of cover and thus does not exploit the correlation among the traffic bytes. The method discloses a mathematical model wherein the traffic byte and the user secret data are processed to obtain the marked encrypted traffic bytes. This mathematical model can overcome the overflow of processed bytes. Further, the data extraction system recovers the secret message and encrypted traffic for further communication to the respective receiving end. At the receiving end, the marked encrypted traffic bytes are processed through a mathematical model to obtain the secret user data and the encrypted channel traffic bytes separately. The proposed method does not require any other communication overhead to process the marked traffic bytes at the receiving end. In detail block diagram is shown in Fig. 4.11.

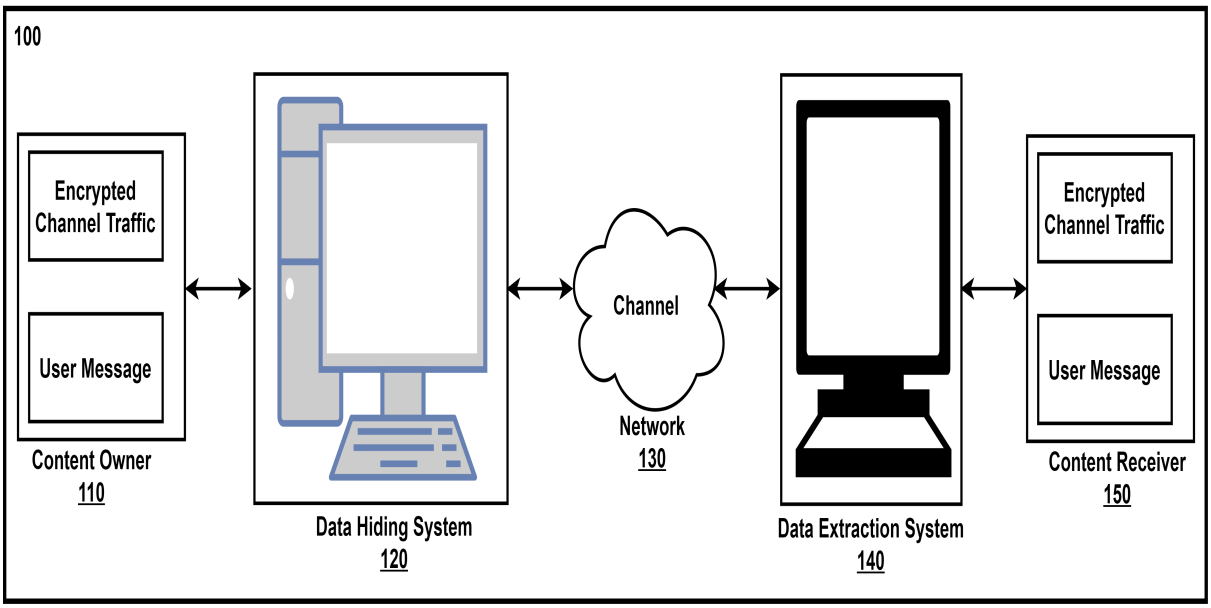

Fig. 4.6 System Overview of Proposed Method

4.5.2 Overview of Data Hiding System

Referring to Fig. 4.7, which gives details of the data hiding system, this system takes encrypted traffic with a secret message as input and generates the encrypted traffic and secret message as output. For hiding the secret message in the encrypted traffic, the hiding unit performs data acquisition through respective interfaces for both the user message and encrypted traffic. The unit comprises Interface, responsible for receiving the input; Memory, which stores the data and embedding parameters; and performs the embedding operations to integrate the secret message into the cover data, producing the marked encrypted traffic. The receiving unit performs data acquisition through the respective interfaces of user message and encrypted traffic. The parameter estimation unit deduces the number of secret message bits to be embedded in the encrypted traffic byte. The secret

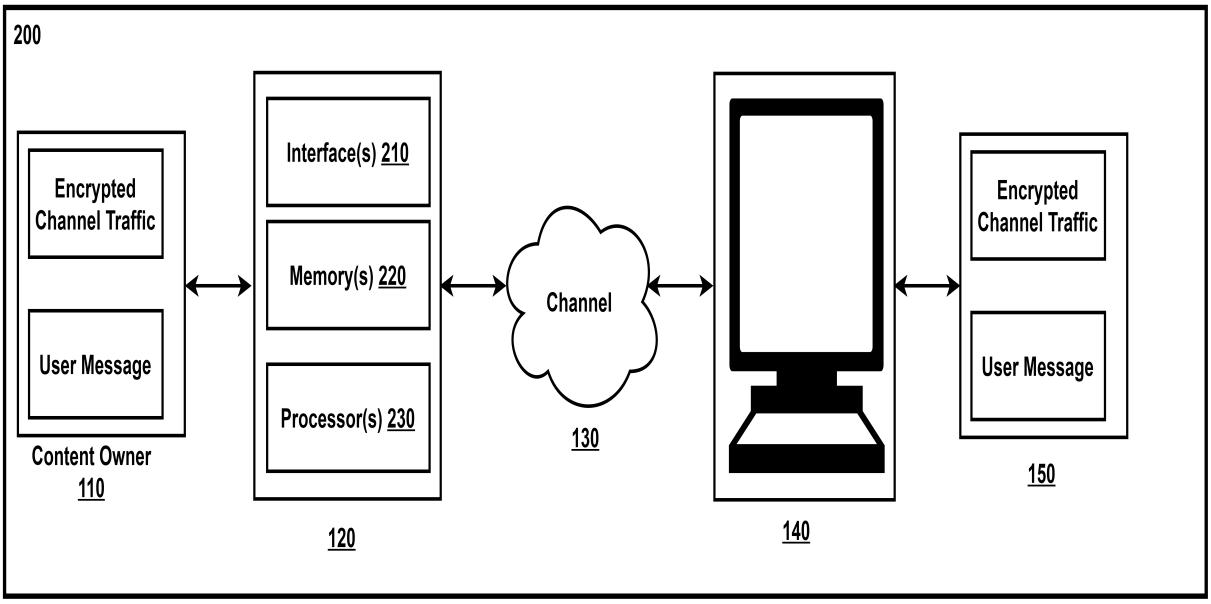

Fig. 4.7 Overview of Data Hiding System

message and the encrypted cover preprocess in the Graphical User Interface(GUI) and are sent to the memory, where there will be an Input Data Source and Other data. From there, it is sent to the Input Unit and then to the Estimation Unit. It's the parameter that

shows how many bits will be embedded in a byte. From there to the Data hiding unit, and then to the transmit unit.

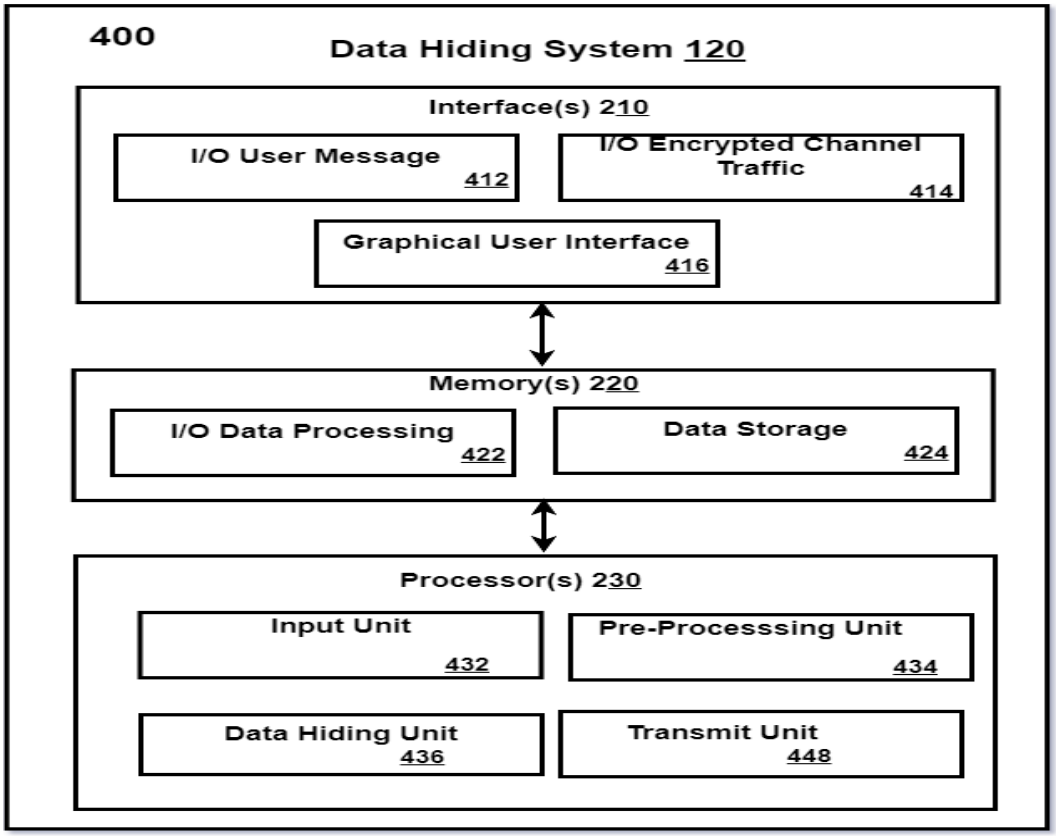

Fig. 4.8 Details Data Hiding System

The proposed system architecture for Reversible Data Hiding (RDH) with an encrypted traffic integrates a comprehensive workflow, enabling secure embedding and transmission of a secret message. The overall process is designed to ensure seamless data hiding while maintaining the integrity and recoverability of the original encrypted traffic. The functional components in Fig. 4.8 and their roles are described below:

- Graphical User Interface (GUI)- The system begins with a user-friendly Graphical User Interface (GUI), which facilitates the pre-processing of the secret message and the encrypted traffic, which is acquired in the acquisition phase by one of the I/O interfaces. The GUI serves as the primary interaction layer where users can input the secret message and the corresponding encrypted traffic. This pre-processing

step ensures the compatibility of both the secret message and the encrypted traffic for subsequent operations. The processed data are then sent to the memory unit for further processing.

- Memory Unit- The memory unit acts as a central repository for data storage and management. It houses two key data sources: Input Data Source: This stores the pre-processed secret message m and the original encrypted traffic received from the GUI. Other Data: Additional metadata or auxiliary information, such as embedding parameters and encoding keys, is stored here for use during the embedding process. The memory unit ensures efficient and organized data retrieval for downstream processing.

- Input Unit -The input unit retrieves the data from the memory unit and prepares it for the embedding process. It serves as an intermediary layer, ensuring that all input data is correctly formatted and synchronized for further operations.

- Estimation Unit -The estimation unit plays a critical role in determining the embedding capacity of the encrypted traffic. It evaluates parameters such as the byte structure and available embedding space to calculate the number of bits that can be embedded in each byte of the encrypted traffic. This unit ensures that the embedding process does not compromise the quality or recoverability of the encrypted traffic.

- Data Hiding Unit -The data hiding unit is the core component responsible for embedding the secret message into the encrypted traffic. Using the parameters determined by the estimation unit, this module encodes the secret message into a format compatible with the encrypted traffic. Embeds the encoded secret data into the encrypted traffic, ensuring minimal distortion and maintaining reversibility. The data hiding unit ensures the integrity of both the embedded data and the original media, enabling

lossless recovery post extraction.

- Transmit Unit -The transmit unit finalizes the process by packaging the modified encrypted traffic for secure transmission. This module ensures that the embedded media is securely transmitted to the receiver while safeguarding it against unauthorized access or tampering. The embedded encrypted traffic is then ready for use in applications requiring secure communication or storage.

4.5.3 Details of Data Extraction System

Referring to Fig. 4.9, which describes the data extraction system. It takes marked encrypted traffic, i.e, encrypted traffic with a secret message as input, and recovers the secret message along with encrypted traffic as output. The receiving unit acquires the marked

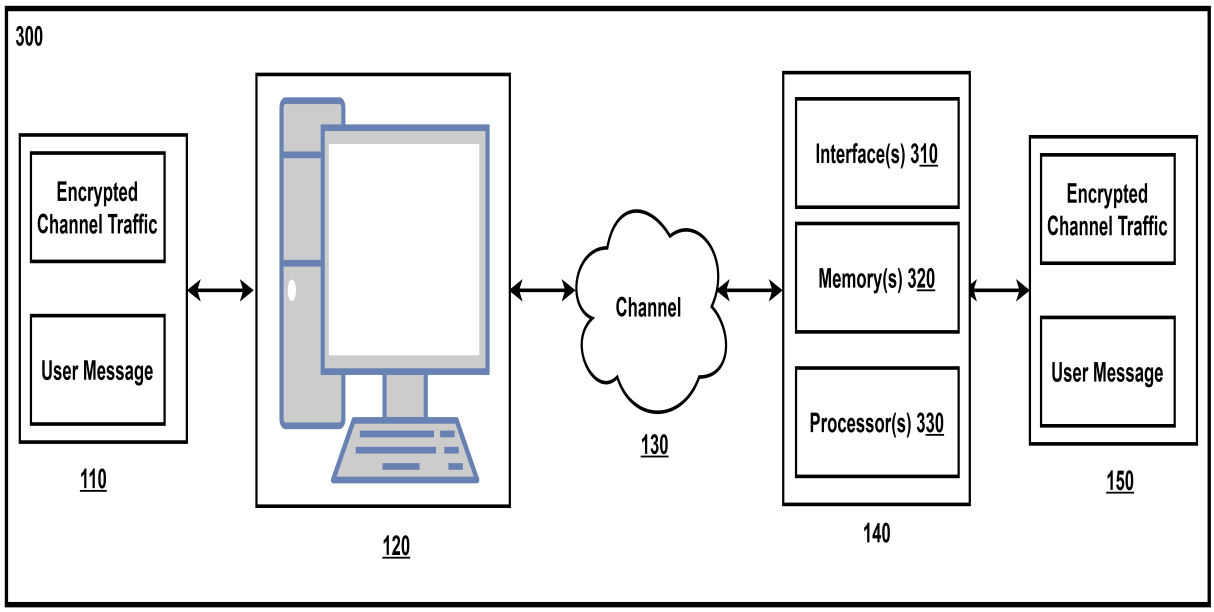

Fig. 4.9 Overview of Data Extraction System

encrypted traffic through the respective interfaces from the sender, transmitted through the communication channel to the receiver. The receiver unit comprises three key components: Interface, which receives the encrypted traffic; Memory, which temporarily stores the received data along with auxiliary information such as decoding keys and embedding

parameters; and Processor, which performs the computations required to segregate the secret message, restore the encrypted traffic to its original state, and recover the embedded message. The parameter estimation unit deduces the information about the number of hidden secret message bits in the encrypted traffic byte. The segregation unit segregates secret and encrypted traffic data. The traffic extraction unit recovers the encrypted traffic, and the message extraction unit recovers the secret message from the segregated data. The functional components in Fig. 4.10 and their roles are described below:

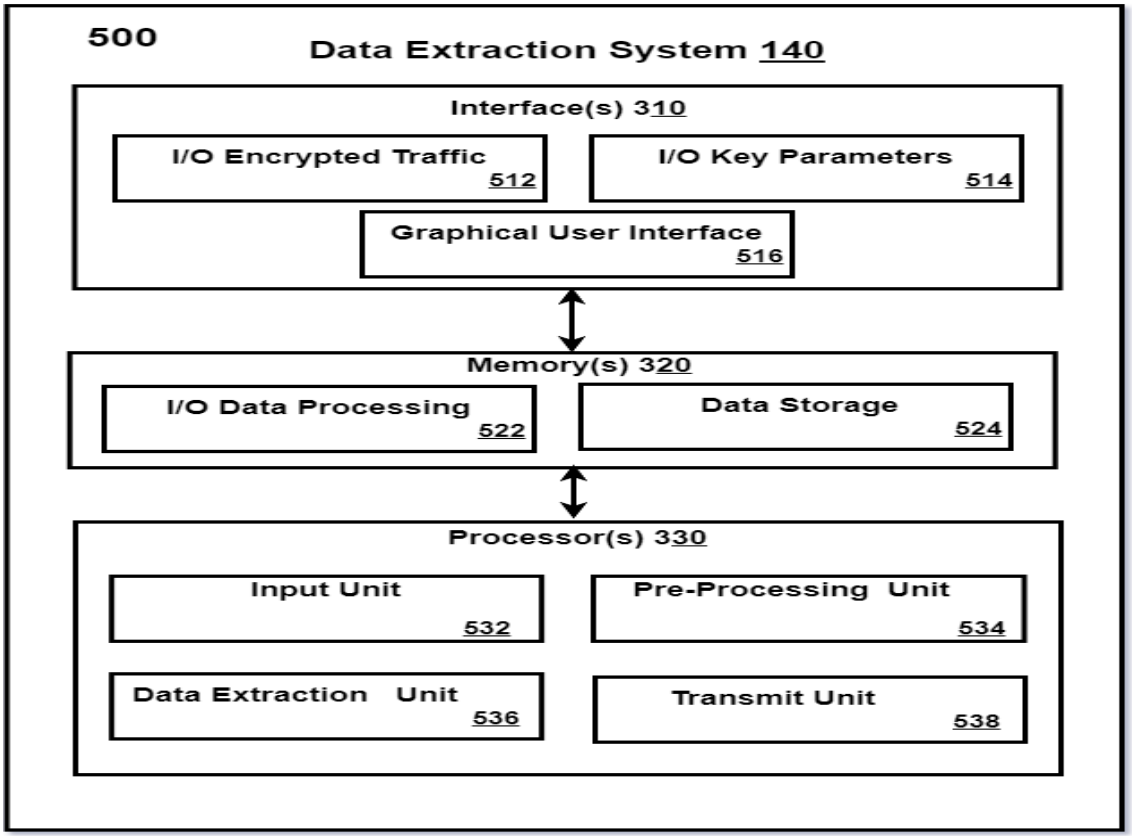

Fig. 4.10 Details of Data Extraction System

- Graphical User Interface (GUI): The system has a user-friendly Graphical User Interface (GUI), which facilitates the pre-processing of the marked encrypted traffic i.e, the encrypted traffic with the secret message. The pre-processing step ensures that the processed marked encrypted traffic is then sent to the memory unit for further processing.

- Memory Unit: The memory unit holds two key data sources: Input Data Source: This stores the pre-processed marked encrypted traffic received from the GUI and, Other Data: Additional metadata or auxiliary information, such as extraction parameters and decoding keys, is stored here for use during the extraction process.

- Receiving unit: The receiving unit retrieves the marked encrypted traffic from the memory unit and prepares it for the extraction process.

- Estimation unit: The estimation unit plays a critical role as it's the parameter that evaluates the number of bits that have been embedded in a byte. It evaluates the parameters, such as the byte structure and available embedding space, to calculate the number of bits that can be embedded in each byte of the encrypted traffic. This unit ensures that the embedding process does not compromise the quality or recoverability of the encrypted traffic. This also holds the information about the number of copies of encrypted traffic used for embedding.

- Segregation unit: In the segregation unit, it segregates secret messages and encrypted traffic data to transmit to the transmit unit.

- Transmit Unit: This module ensures that the secret data and the original encrypted traffic are securely transmitted to the receiver separately while safeguarding them against unauthorized access or tampering. The original encrypted traffic and the secret message are then ready for use in applications requiring secure communication or storage and are retrieved according to their need, whether the secret message or encrypted traffic is to be retrieved first.

Referring to Fig. 4.11, the flow diagram illustrates the process for adaptive user message hiding over the encrypted channel traffic and also the process for the extraction of the user

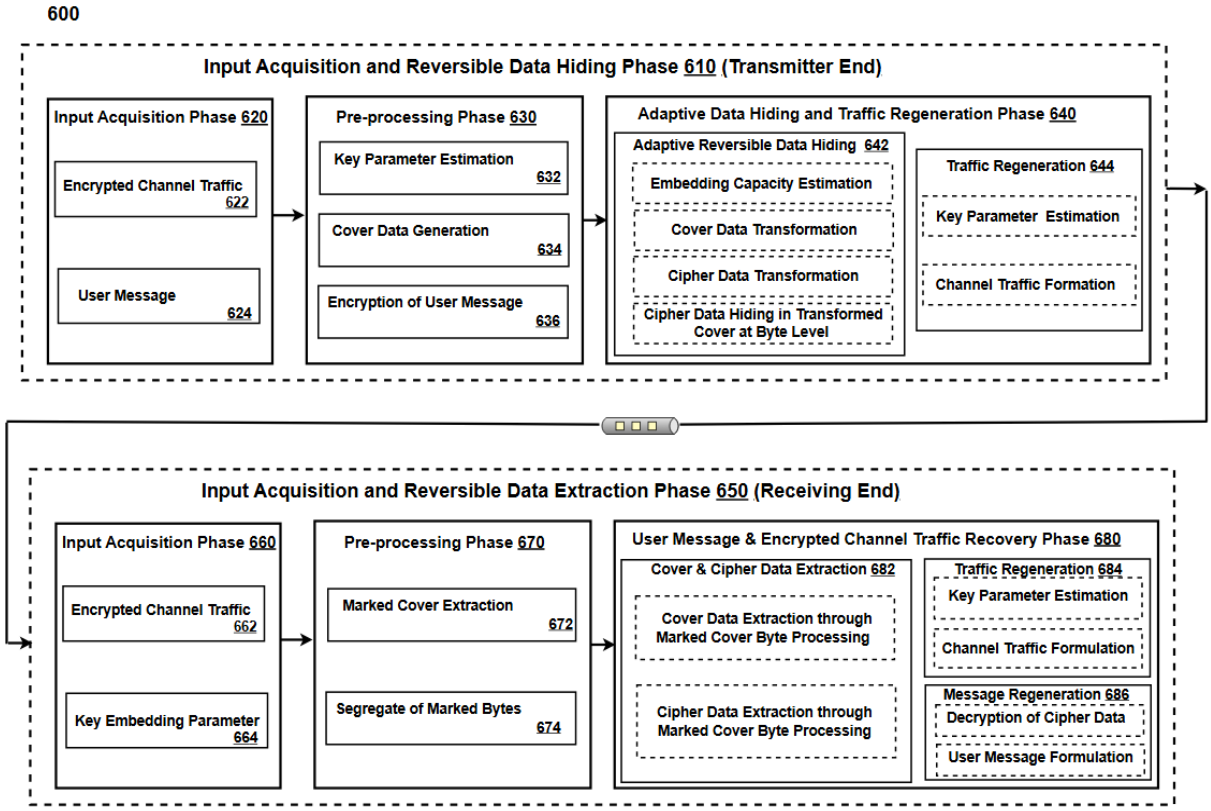

Fig. 4.11 Combined core diagram of data hiding and data extraction

message from the marked encrypted channel traffic. It may consist of input acquisition and the data hiding phase (transmitter end), and input acquisition and the reversible data extraction phase (receiver end). The input phase consists of the two inputs: the encrypted channel traffic and the user message, both of which are subjected to further preprocessing. Further, methods for adaptive user message hiding and reversible data extraction can be mathematically modeled. For this, the processor of the input unit receives the data from the user and the encrypted channel traffic. The user message stream $m = \{m_1, m_2, m_3, ..., m_n\}$, where m_i is the i^{th} frame of the user message stream, is either acquired on real time or stored data which is input to the acquisition stage through the I/O interface. Similarly, encrypted channel traffic $\zeta = \{\zeta_1, \zeta_2, \zeta_3, ..., \zeta_n\}$, where ζ_i is the i^{th} frame of the encrypted channel traffic stream, is acquired in the acquisition phase by one of the I/O interface.

The processor from the preprocessing unit receives the acquired user message m and

encrypted channel traffic ζ and performs key parameter estimation, cover data generation, and encryption of the user message. Input acquired data is first subjected to a protocol dissector wherein the payload is extracted from both the user message and encrypted channel traffic. For this, function f_c extracts the payload from the encrypted channel traffic to obtain the channel payload p_c and key parameters k_c, which mainly include encrypted channel traffic data rate r_c in bps, header, trailer, Mac IDs, and IP address. The i^{th} frame of encrypted channel traffic pre-processing is modelled using Eq. 4.5.1.

$$k_{ci}, p_{ci} = f_c(\zeta_i) \tag{4.5.1}$$

Similarly, function f_u extracts the payload from the user message to obtain the message payload p_u and key parameters k_u, which mainly include the user message payload rate r_u in bps, header, trailer, Mac IDs, and IP address. The i^{th} frame of the user message pre-processing is modeled using Eq. 4.5.2

$$k_{ui}, p_{ui} = f_u(m_i) \tag{4.5.2}$$

Further, cover data generation performs the transformation of the extracted channel payload p_c to obtain the cover data for user message hiding. It involves combining of extracted payload from different frames to obtain a concatenated cover data stream. Cover data stream generation is modeled using Eq. 4.5.3 wherein individual frame payloads are concatenated to obtain the cover data stream.

$$C_s = (p_{c1}, p_{c2}, p_{c3}, ...p_{cn}) \tag{4.5.3}$$

Further, encryption of the user message is performed through the standard block cipher

wherein the user message payload stream $p_u = p_{u1}, p_{u2}, ..., p_{un}$ is transformed through the function f_u to obtain the cipher data C_u. In this, the concatenated user message payload p_u is encrypted using the encryption function to generate cipher data stream C_u as defined in Eq.4.5.4.

$$C_u = f_u(p_u, K) \tag{4.5.4}$$

Where K represents the encryption key used for securing the user message. Cipher data stream C_u is generated before embedding into the encrypted channel traffic payload stream C_s.

The processor in the data hiding unit performs adaptive reversible data hiding. The method for adaptive reversible data hiding consists of embedding capacity estimation, cover data transformation, cipher data transformation, and cipher data hiding in transformed cover at the byte level. In this, for a given channel bandwidth $\mathcal{B}$, maximum channel traffic payload rate $\mathcal{R}_c$ in bps for the channel can be determined through the Shannon Capacity as Eq. 4.5.5.

$$\mathcal{R}_c = \mathcal{B}\log_2(1 + \frac{\mathcal{S}}{\mathcal{N}}) \tag{4.5.5}$$

where $\frac{\mathcal{S}}{\mathcal{N}}$ is the signal to noise ratio. For a noiseless channel, the Nyquist criterion gives the maximum channel capacity $\mathcal{R}_{cm}$ in bps as a theoretical limit, which is determined using Eq. 4.5.6 for M signaling levels.

$$\mathcal{R}_{cm} = 2\mathcal{B}(1 + M) \tag{4.5.6}$$

User message embedding capacity ξ_c is determined, wherein it indicates the number of user message bits to be embedded in the cover data bytes. In this, n user message bits can be embedded into the cover data bytes, provided they satisfy the following condition

as modeled in Eq. 4.5.7.

$$\frac{r_u}{2} \leq (r_c \times \frac{2^n}{8}) \leq \mathcal{R}_c < \mathcal{R}_{cm} \tag{4.5.7}$$

In this, $n = 7$ can be achieved per user with no error in cipher data and cover data recovery. This can further be extended for multiple user data embedding with the condition of First IN and Last OUT sequence for data embedding and data extraction. In case of $\mathfrak{R}$, users' embedding condition is modeled as Eq.4.5.8.

$$\frac{r_{u\mathfrak{R}}}{2} \leq (\mathfrak{R} \times r_c \times \frac{2^{n \times \mathfrak{R}}}{8}) \leq \mathcal{R}_c < \mathcal{R}_{cm} \tag{4.5.8}$$

Where $r_{u\mathfrak{R}}$ represents the combined multiple user message payload rate in bps. Further, cover data transformation is performed wherein the cover data stream C_s is segmented into bytes $\mathfrak{B} = \{\mathfrak{B}_1, \mathfrak{B}_2, \mathfrak{B}_3, ..., \mathfrak{B}_n\}$ which is further used for embedding the cipher data bits that are determined adaptively considering the channel conditions and parameters. A user can embed a maximum of $n (1 \leq n \leq 7)$ bits per byte and at least 2^n copies of each cover byte $\mathfrak{B}_i$ is required for the exact recovery of the user message and original encrypted channel traffic. These copies are further used by another user as a fresh copy for multiple user embedding. In similar lines, cipher data transformation is performed wherein the cipher data stream C_u is segmented into blocks C_b with n bits in each block, depending on the user requirement, using the function f_b. These blocks are further formatted to the numeral number system using the function f_{ns}. These functions are determined using Eq. 4.5.9 and Eq.4.5.10 respectively.

$$C_b = f_b(C_u, n) \tag{4.5.9}$$

$$C_{ns} = f_{ns}(C_b)_{10} \tag{4.5.10}$$

where C_b represents the blocks of cipher data stream with n bits in each block. C_{ns} is

embedded into the transformed cover bytes $\mathfrak{B}$ using function f_{eb} which is modeled using Eq.4.5.11. For instance, i^{th} block of C_b is embedded into the i^{th} byte of $\mathfrak{B}$.

$$\mathfrak{B}_{ms} = f_{eb}(C_{ns}, \mathfrak{B}) \tag{4.5.11}$$

where, $\mathfrak{B}_{ms}$ represents the marked cover byte stream. The marked cover data stream is formed by concatenating the multiple marked cover data stream $C_{ms} = \{\mathfrak{B}_{ms1}, \mathfrak{B}_{ms2}, ..., \mathfrak{B}_{msn}\}$. For instance, in the case of user embedding n bits in the i^{th} transformed cover data byte $\mathfrak{B}_i$, then, 2^n copies $\mathfrak{B}_{ig} = \{\mathfrak{B}_{i1}, \mathfrak{B}_{i2}, ..., \mathfrak{B}_{i2^n}\}$ of the cover data byte $\mathfrak{B}_i$ are generated. In order to overcome the overflow, the method is devised for the generation of copies of transferred cover data bytes wherein transformed cover data byte $\mathfrak{B}_i$ is first divided by 2^n followed by estimation of quotient Q and remainder λ. In this, the remainder may range from 0 to $2^n - 1$.

In case of zero remainder after the division of transformed cover byte $\mathfrak{B}_i$, 2^n copies of transformed cover bytes are assigned to the intermediate values as the quotient Q leading to estimation of $\mathfrak{B}_{igq} = \{Q, Q, ..., Q\}$. Further, each regenerated byte is multiplied by the 2^n to obtain updated bytes as $\mathfrak{B}_{igu} = \{Q \times 2^n, Q \times 2^n, ..., Q \times 2^n\}$ which are used for data embedding. These bytes are further used for the generation of marked encrypted channel payload bytes as $\mathfrak{B}_{imc}$ wherein each byte of the generated cover data is added with the decimal value of the block of cipher data C_{ns}.

In case of non-zero remainder after the division of transformed cover byte $\mathfrak{B}_i$, 2^n copies of transformed cover bytes may be assigned to the intermediate values of the obtained quotient Q plus one for the first λ bytes of $\mathfrak{B}_{ig}$ whereas the remaining bytes of $\mathfrak{B}_{ig}$ are assigned to the values of quotient Q leading to estimation of $\mathfrak{B}_{igq} = \{Q+1, ..., Q, ..., Q\}$. Further, each regenerated byte is multiplied by the 2^n to obtain updated bytes as $\mathfrak{B}_{igu} =$

$\{(Q+1)\times 2^n, ..., Q\times 2^n, ..., Q\times 2^n\}$ which are used for data embedding. These bytes are further used for the generation of marked encrypted channel payload bytes as $\mathfrak{B}_{imc}$ wherein each byte of the generated cover data is added with the decimal value of the block cipher data C_{ns}.

Further, the generated marked encrypted channel payload stream B_{ms} is generated through concatenating the generated marked encrypted channel payload bytes $\mathfrak{B}_{imc}$.

The processor with the transmit unit performs the traffic regeneration, which consists of key parameter estimation and channel traffic formulation. The generated marked cover data stream $\mathfrak{B}_{ms}$ is processed for the generation of corresponding marked encrypted channel traffic using function f_{cr}. The i^{th} frame of the marked encrypted channel traffic ζ_{im} regeneration is modelled using Eq. 4.5.12 wherein the new key parameters k_{nc} mainly include header, trailer, Mac ID's, frame checksum and IP address and these parameters are determined for the formulation of marked encrypted traffic frames.

$$\zeta_{im} = f_{cr}(k_{nci}, \mathfrak{B}_{ms}) \tag{4.5.12}$$

The marked encrypted channel traffic is further transmitted to the receiving end along with key embedding parameters for recovery of the user message and encrypted channel traffic. Further, this method can be extended to multiple users with the condition of a First IN Last OUT sequence for data embedding and data extraction.

In the input acquisition and reversible data extraction phase(receiver end), the input acquisition phase consists of two inputs, that is, the marked encrypted channel traffic and key embedding parameter, both of which are subjected to further processing. The input unit receives the key embedding parameters and marked encrypted channel traffic at the receiving end. The marked encrypted channel traffic $\zeta_m = \{\zeta_{1m}, \zeta_{2m}, \zeta_{3m}, ..., \zeta_{im}\}$ where ζ_{im} is

the i^{th} frame of the marked encrypted channel traffic, is acquired in the acquisition phase by the I/O interface. Similarly, the key embedding parameters such as the total number of cover byte copies, embedding bits per byte $\wp$, and the number of users $\mathfrak{N}$, are required as input to the acquisition stage through the I/O interface.

At the receiving end, the processor with the pre-processing unit receives the marked encrypted channel traffic ζ_m and performs the marked cover extraction and segregation of the marked bytes. For this, using the function f_{mc}, the payload from the marked encrypted channel traffic is extracted to obtain the marked channel payload p_{mc} and key parameters k_{mc}, which mainly include encrypted channel traffic data rate r_{mc}, header, trailer, Mac ID's and IP address. The i^{th} frame of marked encrypted channel traffic pre-processing is modeled using Eq. 4.5.13.

$$k_{mci}, p_{mci} = f_{mc}(\zeta_{im}) \tag{4.5.13}$$

Where k_{mci}, p_{mci} represent the key parameters of the marked encrypted channel traffic and the marked cover payload, respectively. Thus, the marked cover data stream may be obtained by concatenating the individual frames of the marked cover data stream, which is represented using Eq. 4.5.14.

$$C_{ms} = (p_{mc1}, p_{mc2}, ..., p_{mcn}) \tag{4.5.14}$$

Further, segregation of marked bytes is performed, wherein the marked cover data stream is segmented into bytes based on the key embedding parameters, which were received as input using the function $f_{\mathfrak{B}m}$, The segregation of the marked bytes is modeled using the Eq. 4.5.15

$$\mathfrak{B}_{ms} = f_{\mathfrak{B}m}(C_{ms}, 8) \tag{4.5.15}$$

where $\mathfrak{B}_{ms}$ represents the number of marked cover data bytes stream.

The processor with the data extraction unit performs the cover and cipher data extraction. The method for adaptive reversible data extraction, which mainly consists of the cover data extraction through marked cover byte processing and cipher data extraction through marked cover byte processing, are performed. The cover bytes are extracted depending on the number of bits per byte n embedded by the user $\mathfrak{N}$, according to which the number of byte copies $\mathfrak{B}_{ig}$ was generated during embedding. For the recovery of the encrypted traffic, each marked byte $\mathfrak{B}_{mi}$ is divided by the number of copies 2^n and the quotient Q of each $\mathfrak{B}_{mi}$ gives the partial information about the cover bytes. First, whether the quotients are zero or not is checked, and if any, then the zero quotient value is replaced by the quotient value of $\frac{256}{2^n}$, then the corresponding quotient of each $\mathfrak{B}_{mij}$ are added to get the original encrypted traffic byte. For the rest of the non-zero values, the corresponding quotient values are added, and the encrypted traffic is recovered at the receiver end. For the recovery of the cipher message, the remainder λ of each $\mathfrak{B}_{mi}$ gives the cipher message bits. By appending these message bits, the exact cipher message can be recovered for a single user. Similarly, the entire cipher message sent by other users can also be recovered.

The function f_{xc} is performed for the cover data extraction through marked cover byte processing, and this is mathematically modeled using Eq. 4.5.16.

$$p_{xc} = Quotient(f_{xc}(\mathfrak{B}_{ig}, n, \mathfrak{N})) \tag{4.5.16}$$

where, p_{xc} represents the extracted cover data.

Similarly, the cipher data extraction through marked cover byte processing is also performed using the function f_{ed} and is modeled using Eq. 4.5.17

$$C_x = Remainder(f_{ed}(\mathfrak{B}_{ig}, n, \mathfrak{N})) \tag{4.5.17}$$

where C_x represents the cipher message bits.

The processor with the transmit unit performs the traffic regeneration, which consists of key parameter estimation and traffic channel formulation. The key parameters k_c, which mainly include encrypted channel traffic data rate r_c, header, trailer, Mac IDs, and IP address, which were extracted by the sender before embedding, are being estimated and are used further. By embedding the key parameters estimated, the initial encrypted channel traffic $\zeta = \{\zeta_1, \zeta_2, \zeta_3, ..., \zeta_n\}$ is formulated.

And finally, the message regeneration is performed, which consists of the decryption of cipher data to obtain the user message. The cipher is be decrypted using the function f_du, which is modeled using Eq. 4.5.18.

$$p_{ui} = f_{du}(C_{ui}, k) \tag{4.5.18}$$

Further, the decrypted user message blocks are formulated, which correspond to the initial user message stream $m = m_1, m_2, m_3, ..., m_n$.

Here, multiple users can communicate over a common channel without affecting the individual users. Then, secret message bits and the number of users are selected according to the availability of the channel bandwidth. Suppose n-number of users are hiding the user message in the encrypted traffic byte $\mathfrak{B}_i$ sequentially, then the recovery will also be in a sequential manner, thereby enhancing the security and privacy of the user message and

the encrypted channel traffic. It may also be said that the process works based on first-in, last-out. The first marked encrypted channel traffic byte is treated as a new encrypted traffic byte for the next user, and the same process is carried out at the end of the last user. At last, the end user recovered the encrypted byte $\mathfrak{B}_i$ without affecting the end users.

In the Traffic regeneration phase, the key parameters estimated are embedded back into the encrypted traffic, thus formulating the encrypted channel traffic. Simultaneously, the message is also regenerated, wherein the obtained cipher data is decrypted using the AES decryption algorithm, restoring the original user message.

4.6 Conclusion

Most of the separable reversible data hiding methods suffer due to low embedding capacity, which is mainly due to embedding at the pixel level, communication overhead, and overflow/underflow concerns. In addition, embedding capacity depends on the type of cover, which limits its application for specific fields. In the proposed separable RDH method, we achieved high data-hiding capacity by employing bit-pair differences. The use of the average and floor function to generate the scale-up solved the overflow/underflow. The secret message or cover is recovered independently on the basis of available keys rather than in sequential order. In addition, the proposed method is inexpensive as the embedding algorithm only considers simple rule-based data hiding, and data retrieval is achieved by XOR operation. Experimental results revealed a significant improvement in the embedding capacity. Thus, the proposed method performs better qualitatively and quantitatively, while comparison was done with the state-of-the-art methods on different types of cover. When communication over an encrypted channel is considered, most of the communication networks are underutilized as they offer high capacity transmission at multi-gigabit-per-second

data rate. In the proposed method, we hide the user's secret data over an encrypted traffic at the byte level without any communication overhead. The method is independent of the type of cover and does not exploit the correlation among the traffic bytes. Here, embedding is possible multiple times in a single traffic byte, thus enhancing the privacy and security of the data. To overcome the overflow problem, the proposed method uses the modulo 256. At the receiving end, marked encrypted traffic bytes are processed through the mathematical model so as to obtain the secret user data and the encrypted traffic bytes. Thus, the proposed method enhances the user data security and privacy without any degradation of the quality of service.

Chapter 5

Conclusion, Future Scope and Social Impact

In this thesis, novel techniques of multistage and bi-directional Reversible Data Hiding (RDH) for plain cover, Reversible Data Hiding in an Encrypted Image (RDHEI), Separable Reversible Data Hiding in an Encrypted Image (SRDHEI), and Reversible Data Hiding over encrypted traffic have been proposed. For RDH in plain cover, the secret information is embedded in the original cover image, and then, the cover image is encrypted. Overhead of the original cover image is embedded in the encrypted cover image, which reduces the communication time as well as saves the channel bandwidth. To enhance the privacy and security of the cover image and secret information, RDHEI has been investigated. In RDHEI, the extraction of secret information and cover image recovery processes are in sequential order. Which means that secret information is recovered before the cover image. To address this issue, separable RDHEI has been investigated for independent recovery of secret information, and cover image means that the original cover can be recovered without extracting the secret information and vice versa. Further, a data hiding method at the byte level over encrypted traffic without exploiting any correlation between traffic bytes is investigated. In this, multiple users can hide secret data over the encrypted traffic without affecting the communication between end users. This chapter discusses the

proposed techniques on reversible data hiding and also highlights the future directions. The following are the thesis's specific contributions.

Firstly, an exhaustive review of recent developments in the field of reversible data hiding has been performed. For this, reversible data hiding methods are precisely categorized mainly into a) Plain Domain, and b) Encrypted Domain. Further research work is subcategorized, wherein the merits and demerits of each class are highlighted. For instance, RDH work in the field of plain domain is further categorized to a) Lossless Compression, b) Expansion, c) Histogram Shifting, d) Interpolation Techniques. On the other hand, encrypted domain work is classified to: a) Reserving Room after Encryption, b) Reserving Room before Encryption. Performance of state-of-the-art reversible data hiding techniques was analysed via performance metrics, namely embedding capacity, imperceptibility, and computational complexity. In this sequence, the state-of-the-art dual image reversible data hiding methods were also reviewed. These are categorized as exploiting modification direction, pixel value differencing, central folding strategy, and turtle shell, and also explore their merits and demerits. Performance of various dual image reversible data hiding methods is compared to in the context of data concealing capacity and visual quality of stego images. Additionally, a comprehensive security analysis of data hiding techniques using different attacks and their impact on hidden data and cover media was also thoroughly studied. These attacks are classified as histogram equalization, filtering, contrast enhancement, denoising, compression, geometric transformation, and malicious attacks.

Most of the RDH techniques hide the overhead bits in the same domain of the cover image, which leads to a reduction in embedding capacity. For this, a multistage high-capacity reversible data hiding technique without overhead is proposed, in which the data hiding capacity is increased by hiding the secret message in plain cover and overhead information in the encrypted cover. In this, the reversible data hiding approach exploits histogram

peaks for embedding the secret data along with overhead bits both in plain and encrypted domains. First, a marked image is obtained by embedding secret data in the plain domain, which is further processed using the affine cipher, maintaining correlation among the pixels. In the second stage, overhead bits are embedded in the encrypted marked image. High embedding capacity is achieved through exploiting the histogram peak for embedding multiple bits of secret data.

As an extension, bidirectional embedding of secret data is proposed wherein both left peak and right peak from the main histogram peak are considered for secret data embedding, thereby achieving high embedding capacity. In addition, multistage embedding both at the plain domain and encrypted domain not only improved the data security but also overcame the need for a separate communication channel.

Most of the separable reversible data hiding methods suffer due to low embedding capacity, which is mainly due to embedding at the pixel level, communication overhead, and overflow/underflow concerns. In addition, embedding capacity depends on the type of cover, which limits their application for specific fields. For this, a separable RDH method is proposed to achieve high data-hiding capacity by employing bit-pair differences. Embedding at bit-pair level not only achieves high secret data embedding capacity but also enhances the security and privacy of the secret message and cover. The proposed method employs the average and floor function to generate the interpolated cover and hence, avoids the overflow/underflow problem. This method supports blind data extraction from the marked cover, thereby being computationally inexpensive. In this, secret messages and covers are recovered independently on the basis of available keys rather than sequence order. This method is computationally inexpensive as the embedding algorithm only considers simple rules-based data hiding, and data retrieval is achieved through a simple XOR operation, which overcomes the communication overhead.

Secure communication of sensitive data over the public network is very critical, especially in the private/public 5G/6G cellular network with tremendous deployment of the Internet of Things. Generally, this is achieved either by encrypting plain data before transmission or by hiding data under the cover. On the other hand, most of the communication networks, especially 5G/6G cellular networks, are underutilized as they offer high-capacity transmission at multi-gigabit-per-second data rates. Hence, effective utilization of the communication channel along with security and privacy of users is paramount for a channel dedicated to the Internet of Things or Military Applications. Nowadays, most of the traffic over the communication channel is encrypted, and applications ensure end-to-end encryption of user data. Accordingly, solutions for data hiding over an encrypted channel are invented that exploit redundant information or correlation among blocks of encrypted data for single user secure data hiding. This led to the underutilization of the communication channel and other resources. In addition, existing methods for reversible data hiding could not achieve high embedding capacity due to less correlation in the encrypted traffic over the channel and the necessity of communication overhead for faithful recovery of secret data and cover. For this, a method for data hiding at the byte level over encrypted traffic without exploiting any correlation between traffic bytes is proposed, in which multiple users can hide secret data over the encrypted traffic without affecting the communication between end users. This ensures the faithful recovery of cover at each user, efficiently utilizes channel resources, and also ensures privacy and security of multi-user data transmission over a common channel.

5.1 Future Directions

As the field advances, several avenues for future exploration remain, promising to expand the potential of RDH. The research work in the field of reversible data hiding can be further extended in the following future directions.

- Attacks using deep learning methods can be investigated for the development of robust solutions. Methods should also cater for the optimal capacity and tunable quality of the cover image.

- Generalized framework for both encrypted and plain domains with application of AI methods can be investigated for achieving optimal utilization of the cover media and communication channel.

- Integrating machine learning algorithms can help resolve the trade-off between the type of cover media and the embedding capacity, ensuring more efficient and dynamic embedding processes.

- Research in RDH should be focused on the development of robust and adaptive embedding algorithms that can withstand various attacks and enhance the security and resilience of RDH techniques.

- Bidirectional multistage technique can be extended for multimedia applications to increase the embedding capacity. Exploiting colour images for data embedding can be explored further to achieve multidimensional embedding of secret data.

- Exploring ways to develop the RDH techniques independent of the cover image quality is another direction to be explored in the future.

- Bitpair embedding offers an intriguing direction by utilizing cyclic embedding techniques, where bitpairs can be employed in a cyclic order to optimize data embedding.

5.2 Social Impact

The social impact of reversible data hiding (RDH) is profound and multifaceted, spanning diverse sectors such as privacy, intellectual property, healthcare, digital forensics, and cultural preservation. RDH serves as a cornerstone for secure and innovative digital interactions, allowing sensitive information to be embedded, shared, and retrieved without compromising the original content. This capability empowers industries to strike a balance between security and accessibility, fostering trust and collaboration in an increasingly interconnected world.

Nevertheless, the technology comes with its own set of challenges. The potential for misuse, such as concealing malicious or unethical activities, underscores the need for robust safeguards, ethical guidelines, and regulatory oversight. Addressing these challenges requires a proactive approach, including investments in research, user education, and the development of detection tools to prevent malicious applications of RDH.

Looking forward, the adoption of RDH is poised to grow alongside advancements in artificial intelligence, blockchain, and quantum computing. These synergies can unlock new opportunities for RDH to revolutionize data management, enhance creative industries, and support sustainable practices. Nevertheless, to ensure its positive societal impact, stakeholders—including technologists, policymakers, and users—must collaborate to establish a framework that prioritizes ethical use, inclusivity, and transparency.

In conclusion, reversible data hiding is not just a technical innovation but a societal enabler that holds the potential to redefine how we interact with and protect digital content. By

leveraging its strengths responsibly and addressing its risks proactively, RDH can serve as a transformative force for good in the digital age.

References

[1] J. Barton, “Method and apparatus for embedding authentication information within digital data,” *U.S. Patent,5,646,997*, 1997.

[2] J. Fridrich, M. Goljan, and R. Du, “Lossless data embedding-new paradigm in digital watermarking,” *EURASIP J. Adv. Signal Process*, vol. 2002, no. 2, pp. 185–196, 2002.

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

[4] J. Fridrich, M. Goljan, and R. Du, “Invertible authentication,” *Proc. SPIE, Security and Watermarking of Multimedia Contents*, vol. 4314, pp. 197–208, 2001.

[5] J. Tian, “Wavelet-based reversible watermarking for authentication,” *Proceedding SPIE*, vol. 4675, pp. 679–690, 2002.

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

[7] A. M. Alattar, “Reversible watermark using the difference expansion of a generalized

integer transform," *IEEE Transactions on Image Processing*, vol. 13, no. 8, pp. 1147–1156, 2004.

[8] D. M. Thodi and J. J. Rodriguez, "Expansion embedding techniques for reversible watermarking," *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 721–730, 2007.

[9] Zhicheng Ni, Yun-Qing Shi, N. Ansari, and Wei Su, "Reversible data hiding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 3, pp. 354–362, 2006.

[10] C. De Vleeschouwer, J. E. Delaigle, and B. Macq, "Circular interpretation of histogram for reversible watermarking," in *2001 IEEE Fourth Workshop on Multimedia Signal Processing (Cat. No.01TH8564)*, 2001, pp. 345–350.

[11] S. Lee, Y. Suh, and Y. Ho, "Reversible image authentication based on watermarking," in *2006 IEEE International Conference on Multimedia and Expo*, 2006, pp. 1321–1324.

[12] V. M. Manikandan and P. Renjith, "An efficient overflow handling technique for histogram shifting based reversible data hiding," in *2020 International Conference on Innovative Trends in Information Technology (ICITIIT)*, 2020, pp. 1–6.

[13] C. Shaji and S. Sam, "A new data encoding based on maximum to minimum histogram in reversible data hiding," *The Imaging Journal*, vol. 67, no. 4, pp. 202–214, 2019.

[14] J. Wang, X. Chen, J. Ni, N. Mao, and Y. Shi, "Multiple histograms based reversible data hiding: Framework and realization," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2019.

[15] K. Jung and K. Yoo, “Data hiding method using image interpolation,” *Computer Satandards and Interfaces*, vol. 31, no. 2, pp. 465–470, 2009.

[16] D. Loua, C. Choub, H. Weia, and H. Huang, “Active steganalysis for interpolation -error based reversible data hiding,” *Pattern Recognition Letters*, vol. 34, no. 9, pp. 1032–1036, 2013.

[17] M. Khosravi and M. Yazdi, “A lossless data hiding scheme for medical images using a hybrid solution based on ibrw error histogram computation and quartered interpolation with greedy weights,” *Neural Computing and applications*, vol. 30, pp. 2017–2028, 2018.

[18] M. A. Wahed and H. Nyeem, “Efficient lsb substitution for interpolation based reversible data hiding scheme,” in *2017 20th International Conference of Computer and Information Technology (ICCIT)*, 2017, pp. 1–6.

[19] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, “Reversible data hiding in encrypted images by reserving room before encryption,” *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 3, pp. 553–562, 2013.

[20] T. Mathew and M. Wilscy, “Reversible data hiding in encrypted images by active block exchange and room reservation,” in *2014 International Conference on Contemporary Computing and Informatics (IC3I)*, 2014, pp. 839–844.

[21] W. Z. K. Ma and N. Yu, “Reversibility improved data hiding in encrypted images,” *Signal Processing*, vol. 94, pp. 118–127, 2013.

[22] Y. Puyang, Z. Yin, and Z. Qian, “Reversible data hiding in encrypted images with two-msb prediction,” in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2018, pp. 1–7.

[23] X. Zhang, “Reversible data hiding in encrypted image,” *IEEE Signal Processing Letters*, vol. 18, no. 4, pp. 255–258, 2011.

[24] X.Zhang, “Separable reversible data hiding in encrypted image,” *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 826–832, 2012.

[25] J. Yu, G. Zhu, X. Li, and J. Yang, “An improved algorithm for reversible data hiding in encrypted image,” in *The International Workshop on Digital Forensics and Watermarking 2012. IWDW 2012*, vol. LNCS 7809(3), 2016, pp. 441–452.

[26] L. Kamstra and H. J. A. M. Heijmans, “Reversible data embedding into images using wavelet techniques and sorting,” *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 2082–2090, 2005.

[27] D. M. Thodi and J. J. Rodriguez, “Prediction-error based reversible watermarking,” in *2004 International Conference on Image Processing, 2004. ICIP ’04.*, vol. 3, 2004, pp. 1549–1552 Vol. 3.

[28] Y. Hu, H. Lee, and J. Li, “De-based reversible data hiding with improved overflow location map,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 2, pp. 250–260, 2009.

[29] H. Zhang, C. Wang, J. Wang, and S. Xiang, “A new reversible watermarking scheme using the content-adaptive block size for prediction,” *Signal Processing*, vol. 164, pp. 74–83, 2019.

[30] C. Honsinger, P. Jones, M. Rabbani, and J. Stffel, “Lossless recovery of an original image containing embedded data,” *U.S. Patent,6,278,791*, 2001.

[31] G. Xuan, J. Zhu, J. Chen, Y. Shi, Z. Ni, and W. Su, “Distortionless data hiding based

on integer wavelet transform," *IEEE, Journal ELECTRONICS LETTERS*, vol. 38, no. 25, pp. 1646–1648, 2002.

[32] I.Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *J. Fourier Anal. Appl.,*, vol. 4, pp. 247–269, 1998.

[33] I. W. M. Radford and J. Cleary, "Arithmetic coding for data compression," *Commun. ACM*, vol. 30, no. 6, pp. 520–540, 1987.

[34] L.S.TChen, S. Lin, and J. C. Lin, "Reversible jpeg-based hiding method with high hiding-ratio," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 24, no. 3, pp. 433–456, 2010.

[35] J. He, X. Pan, H. T. Wu, and S. Tang, "Improving block ordering and frequency selection for reversible data hiding in jpeg images," *Signal Processing*, vol. 175, 2020.

[36] T. Kalker and F. M. J. Willems, "Capacity bounds and constructions for reversible data-hiding," in *2002 14th International Conference on Digital Signal Processing Proceedings. DSP 2002 (Cat. No.02TH8628)*, vol. 1, 2002, pp. 71–76 vol.1.

[37] A. Malik, G. Sikka, and H. Verma, "An ambtc compression based data hiding scheme using pixel value adjusting strategy," *Multidim Syst Sign Process*, vol. 29, no. 2, pp. 1801–1818, 2018.

[38] W. Zhang, B. Chen, and N. Yu, "Capacity-approaching codes for reversible data hiding," *Lecture Notes in Computer Science, Springer*, vol. 6958, pp. 255–269, 2011.

[39] S. Lin and W. Chung, "The scalar scheme for reversible information-embedding in gray-scale signals: Capacity evaluation and code constructions," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 4, pp. 1155–1167, 2012.

[40] J. Lin, S. Weng, T. Zhang, B. Ou, and C. Chang, “Two-layer reversible data hiding based on ambtc image with (7, 4) hamming code,” *IEEE Access*, vol. 8, pp. 21 534–21 548, 2020.

[41] S. Lee, C. D. Yoo, and T. Kalker, “Reversible image watermarking based on integer-to-integer wavelet transform,” *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 3, pp. 321–330, 2007.

[42] Y. Hu, H. Lee, K. Chen, and J. Li, “Difference expansion based reversible data hiding using two embedding directions,” *IEEE Transactions on Multimedia*, vol. 10, no. 8, pp. 1500–1512, 2008.

[43] O. Quershi and B.EKhoo, “Two dimensional difference expansin (2d-de) scheme with a characteristics based threshold,” *Signal Processing*, vol. 93, pp. 154–162, 2013.

[44] C. chen, Y. Tsai, and H. yeh, “Difference expansion based reversible and visible image watermarking scheme,” *Multimedia tools*, vol. 10, no. 8, pp. 1500–1512, 2016.

[45] A. Arham, H. Nugroha, and T. Adji, “Multiple layer data hiding scheme based on difference expansion of quad,” *Signal Processing*, vol. 137, pp. 52–62, 2017.

[46] M. U. Celik, G. Sharma, A. M. Tekalp, and E. Saber, “Reversible data hiding,” in *Proceedings. International Conference on Image Processing*, vol. 2, 2002, pp. 157–160.

[47] V. Sachnev, H. J. Kim, J. Nam, S. Suresh, and Y. Q. Shi, “Reversible watermarking algorithm using sorting and prediction,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 7, pp. 989–999, 2009.

[48] D. Coltuc, “Improved embedding for prediction-based reversible watermarking,” *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 873–882, 2011.

[49] A. Weber, “The usc-sipi image database:version 4,” *International Journal of Pattern Recog. and Artif. Intell.*, vol. 28, no. 03, 2014.

[50] R. Uyyala and R. Pal, “Reversible data hiding using improved gradient based prediction and adaptive histogram bin shifting,” in *2020 7th International Conference on Signal Processing and Integrated Networks (SPIN)*, 2020, pp. 720–726.

[51] S. Yang, “Saliency-based image contrast enhancement with reversible data hiding,” in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 2847–2851.

[52] X. Li, B. Yang, and T. Zeng, “Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection,” *IEEE Transactions on Image Processing*, vol. 20, no. 12, pp. 3524–3533, 2011.

[53] D. Coltuc, “Low distortion transform for reversible watermarking,” *IEEE Transactions on Image Processing*, vol. 21, no. 1, pp. 412–417, 2012.

[54] X. Li, J. Li, and B. yang, “High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion,” *Signal Processing*, vol. 93, no. 1, pp. 198–205, 2013.

[55] F. Peng, X. Li, and B. yang, “Improved ivo-based reversible data hiding,” *Digital Signal Processing*, vol. 25, pp. 255–265, 2013.

[56] N. Jain and S. Kasana, “High capacity reversible data hiding using modified pixel value ordering approach,” *Journal of Circuits, Systems and Computers*, vol. 27, no. 11, 2018.

[57] M. Kumar and S. Agarwal, “Reversible data hiding based on prediction error expan-

sion using adjacent pixels," *Security and communication Networks*, vol. 9, no. 16, pp. 574–578, 2016.

[58] B. Ou, X. Li, Y. Zhao, R. Ni, and Y. Shi, "Pairwise prediction-error expansion for efficient reversible data hiding," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 5010–5021, 2013.

[59] B. Ou, X. Li, Y. Zhao, and R. Ni, "Reversible data hiding using invariant pixel-value-ordering and prediction-error expansion," *Signal Processing*, vol. 29, no. 7, pp. 760–772, 2014.

[60] R. Abbasi, N. Qureshi, and H. Hassan, "Generalized pvo-based dynamic block reversible data hiding for secure transmission using firefly algorithm," *Transaction on Emerging Telecommunications Technologies*, 2019.

[61] R. Franzen, "Kodak lossless true color image suite," *http://www.r0k.us/graphics/kodak*, vol. 28, no. 03, 2014.

[62] P. Bas, T. Filler, and T. Pevny, "Break our steganographic system": The ins and outs of organizing boss," in *IH 2011: Information Hiding*, vol. LNCS, 6958, 2011, pp. 59–70.

[63] R. Kumar and K. Jung, "Robust reversible data hiding scheme based on two-layer embedding strategy," *Information Sciences*, vol. 512, pp. 96–107, 2020.

[64] N. Li and F. Huang, "Reversible data hiding for jpeg images based on pairwise nonzero ac coefficient expansion," *Signal Processing*, vol. 171, 2020.

[65] M. Fallahpour and M. Sedaaghi, "High capacity lossless data hiding based on histogram modification," *IEICE Electronics Express*, vol. 4, no. 7, pp. 205–210, 2007.

[66] C. Lin, W. Tai, and C. Chang, “Multilevel reversible data hiding based on histogram modification of difference images,” *Pattern Recognition*, vol. 41, no. 12, pp. 3582–3591, 2008.

[67] W. Tai, C. Yeh, and C. Chang, “Reversible data hiding based on histogram modification of pixel differences,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 6, pp. 906–910, 2009.

[68] X. Li, B. Li, B. Yang, and T. Zeng, “General framework to histogram-shifting-based reversible data hiding,” *IEEE Transactions on Image Processing*, vol. 22, no. 6, pp. 2181–2191, 2013.

[69] M. Fujiyoshi, S. Sato, H. L. Jin, and H. Kiya, “A location-map free reversible data hiding method using block-based single parameter,” in *2007 IEEE International Conference on Image Processing*, vol. 3, 2007, pp. III – 257–III – 260.

[70] M. Fujiyoshi, T. Tsuneyoshi, and H. Kiya, “A parameter memorization-free lossless data hiding method with flexible payload size,” *IEICE Electronics Express*, vol. 7, no. 23, pp. 1702–1708, 2010.

[71] B. Ou and Y. Zhao, “High capacity reversible data hiding based on multiple histograms modification,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 8, pp. 2329–2342, 2019.

[72] X. Li, W. Zhang, X. Gui, and B. Yang, “Efficient reversible data hiding based on multiple histograms modification,” *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 2016–2027, 2015.

[73] W. Qi, X. Li, T. Zhang, and Z. Guo, “Optimal reversible data hiding scheme based

on multiple histograms modification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 8, pp. 2300–2312, 2019.

[74] P. Garg, S. Kasana, and G. Kasana, "Block- based reversible data hiding using histogram shifting and modulus operator for digital images," *Journal of Circutits, Systems and Computers*, vol. 26, no. 6, pp. 1 750 103–1 750 120, 2017.

[75] X. Gao, L. An, Y. Yuan, D. Tao, and X. Li, "Lossless data embedding using generalized statistical quantity histogram," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 8, pp. 1061–1070, 2011.

[76] X. Li, W. Zhang, X. Gui, and B. Yang, "A novel reversible data hiding scheme based on two-dimensional difference-histogram modification," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 7, pp. 1091–1100, 2013.

[77] X. He, W. Zhang, H. Zhang, L. Ma, and Y. Li, "Reversible data hiding for high dynamic range images using edge information," *Multimedia Tools Applied*, pp. 1–24, 2018.

[78] X. Gao, Z. Pan, E. Gao, and G. Fan, "Reversible data hiding for high dynamic range images using two-dimensional prediction -error histogram of the second time prediction," *Signal Processing*, vol. 173, 2020.

[79] Y. Jia, Z. Yina, X. Zhang, and Y. Luo, "Reversible data hiding based on reducing invalid shifting of pixels in histogram shiftin," *Signal Processing*, vol. 163, pp. 238–246, 2019.

[80] W. Wang and W. Wang, "Hs-based reversible data hiding scheme using median prediction error," *Multimedia Tools Applied*, vol. 79, pp. 18 143–18 165, 2020.

[81] J. He, J. Chen, and S. Tang, "Reversible data hiding in jpeg images based on neg-

ative influence models," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2121–2133, 2020.

[82] Y. Yalman, F. Akar, and I. Erturk, "An image interpolation based reversible data hiding method using r-weighted coding," in *2010 13th IEEE International Conference on Computational Science and Engineering*, 2010, pp. 346–350.

[83] Y. Chang, C. Huang, C. Lee, and S. Wang, "Image interpolating based data hiding in conjunction with pixel-shifting of histogram," *The Journal of Supercomputing*, vol. 66, pp. 1093–1110, 2013.

[84] C. Lee and Y. Huang, "An efficient image interpolation increasing payload in reversible data hiding," *Expert Systems with Applications*, vol. 39, no. 8, pp. 6712–6719, 2012.

[85] M. Jana and B. Jana, "High capacity data hiding using 2d parabolic interpolation," *Computional Intelligence in Pattern Recognition*, vol. 999, pp. 157–169, 2020.

[86] Y. Tsai, J. Chen, Y. Kuo, and C. Chan, "A generalized image interpolation -based reversible data hiding scheme with high embedding capacity and image quality," *KSSI Trans. Internet and Information Sys.*,, vol. 8, no. 9, 2014.

[87] A. Malik, G. Sikka, and H. Verma, "A reversible data hiding scheme for interpolated images based on pixel intensity range," *Multimedia Tools Applied*, vol. 79, 2020.

[88] A. Shaik and T. V, "High capacity reversible data hiding using 2d parabolic interpolation," *Multimedia Tools Applied*, vol. 78, pp. 9717–9735, 2019.

[89] J. Hu and T. Li, "Reversible steganography using extended image interpolation technique," *Computers and Electrical Engineering*, vol. 46, pp. 447–455, 2015.

[90] X. Zhang, Z. Sun, Z. Tang, C. Yu, and X. Wang, “High capacity data hiding based on interpolated image,” *Multimedia Tools Applied*, vol. 76, pp. 9195–9218, 2017.

[91] A. Malik, G. Sikka, and H. Verma, “An image interpolation based reversible data hiding scheme using pixel value adjusting feature,” *Multimedia Tools Applied*, vol. 76, pp. 13 025–13 046, 2017.

[92] M. C. W. Puech and o. Strauss, “A reversible data hiding method for encrypted images,” in *SPIE*, vol. 6819, 2008.

[93] W. Hong, T. Chen, and H. Wu, “An improved reversible data hiding in encrypted images using side match,” *IEEE Signal Processing Letters*, vol. 19, no. 4, pp. 199–202, 2012.

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

[95] F. Di, F. Huang, M. Zhang, J. Liu, and X. Yang, “Reversible data hiding in encrypted images with high capacity by bit-plane operations and adaptive embedding,” *Multimedia Tools and Applications*, vol. 77, pp. 20 917–20 935, 2018.

[96] F. Huang, J. Huang, and Y. Shi, “New framework for reversible data hiding in encrypted domain,” *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 12, pp. 2777–2789, 2016.

[97] X. Wu and W. Sun, “High-capacity reversible data hiding in encrypted images by prediction error,” *Signal Processing*, vol. 104, pp. 387–400, 2014.

[98] R. Zhang, C. Lu, and J. Liu, “A high capacity reversible data hiding scheme for

encrypted covers based on histogram shifting," *Journal of Information Security and Applications*, vol. 47, pp. 199–207, 2019.

[99] Z. Qian, X. Zhang, and G. Feng, "Reversible data hiding in encrypted images based on progressive recovery," *IEEE Signal Processing Letters*, vol. 23, no. 11, pp. 1672–1676, 2016.

[100] G. Kumaresan and N. Gopalan, "Reversible data hiding in encrypted images using public cloud and cellular automata," *Journal of Applied Security Research*, vol. 15, pp. 427–444, 2020.

[101] X. Zhang, C. Qin, and G. Sun, "Reversible data hiding in encrypted images using pseudorandom sequence modulation," in *IWDW 2012: The International Workshop on Digital Forensics and Watermarking 2012*, vol. LNCS, 25, 2013, pp. 358–367.

[102] Y.C.Chen, C.W.Shiu, and G.Horng, "Encrypted signal- based reversible data hiding with public key cryptosystem," *Journal of Visual Communication and Image Representation*, vol. 25, no. 5, pp. 1164–1170, 2014.

[103] S. Zheng, D. Li, D. Hu, D. Ye, L. Wang, and J. Wang, "Lossless data hiding algorithm for encrypted images with high capacity," *Multimedia Tools and Applications*, vol. 75, pp. 13 765–13 778, 2016.

[104] Z. Y. A. A. J. X. Zhang and B. Luo, "Reversible data hiding in encrypted images based on multi-level encryption and block histogram modification," *Multimedia Tools and Applications*, vol. 76, no. 3, pp. 3899–3920–13 778, 2017.

[105] H. L. Q. li, B. Yan and N. Chen, "Separable reversible data hiding in encrypted images with improved security and capacity," *Multimedia Tools and Applications*, vol. 77, no. 4, pp. 30 749–30 768, 2018.

[106] Y. R. Z. Qian, X. Zhang and G. Feng, “Block cipher based separable reversible data hiding in encrypted images,” *Multimedia Tools and Applications*, vol. 75, no. 3, pp. 13 749–13 763, 2016.

[107] D.Xiao and S. Chen, “Separable data hiding in encrypted image based on compressive sensing,” *Electronics Letters*, vol. 50, no. 8, pp. 598–600, 2014.

[108] N. Zhou, M. Zhang, H. Wang, Y. Ke, and F. Di, “Separable reversible data hiding scheme in homomorphic encrypted domain based on ntru,” *IEEE Access*, vol. 8, pp. 81 412–81 424, 2020.

[109] Z. Qian and X. Zhang, “Reversible data hiding in encrypted images with distributed source encoding,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 4, pp. 636–646, 2016.

[110] K. Z. J. Liu and R. Zhang, “A fully reversible data hiding scheme in encrypted images based on homomorphic encryption and pixel prediction,” *Circuits, Systems, and Signal Processing*, vol. 39, pp. 3532–3552, 2020.

[111] S. Xiang and X. Luo, “Reversible data hiding in homomorphic encrypted domain by mirroring ciphertext group,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 11, pp. 3099–3110, 2018.

[112] M. Li and Y. Li, “Histogram shifting in encrypted images with public key cryptosystem for reversible data hiding,” *Signal Processing*, vol. 130, pp. 190–196, 2017.

[113] Z. T. S. X. H. Y. C. Qin and X. Zhang, “Reversible data hiding with differential compression in encrypted image,” *Multimedia Tools and Applications*, vol. 78, pp. 9691–9715, 2019.

[114] Y. Wang, Z. Cai, and W. He, “High capacity reversible data hiding in encrypted image based on intra-block lossless compression,” *IEEE Transactions on Multimedia*, 2020.

[115] C. Jiang and Y. Pang, “Encrypted images-based reversible data hiding in paillier cryptosystem,” *Multimedia Tools and Applications*, vol. 79, pp. 693–711, 2020.

[116] Y. Ke, M. Zhang, J. Liu, T. Su, and X. Yang, “Fully homomorphic encryption encapsulated difference expansion for reversible data hiding in encrypted domain,” *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2019.

[117] D. X. Y. Z. M. Li and H. Nan, “Reversible data hiding in encrypted images using cross division and additive homomorphism,” *Signal Processing: Image Communication*, vol. 39, pp. 234–248, 2015.

[118] Y. Z. S. Yi and Z. Hua, “Reversible data hiding in encrypted images using adaptive block-level prediction-error expansion,” *Signal Processing*, vol. 64, pp. 78–88, 2018.

[119] D. H. W. Zhang, H. Wang and N. Yu, “Reversible data hiding in encrypted images by reversible image transformation,” *IEEE Transactions on Multimedia*, vol. 18, no. 8, pp. 1469–1479, 2016.

[120] Y.-L. Lee and W.-H. Tsai, “A new secure image transmission technique via secret-fragment-visible mosaic images by nearly reversible color transformations,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 4, pp. 695–703, 2014.

[121] Z. Liu and C. Pun, “Reversible data hiding in encrypted images by redundant space transfer,” *Information Sciences*, vol. 433, pp. 188–203, 2018.

[122] W. H. C.Qin, X. Qian and X. Zhang, “An efficient coding scheme for reversible data

hiding in encrypted image with redundancy transfer," *Information Sciences*, vol. 487, pp. 176–192, 2019.

[123] Y. Fu, P. Kong, H. Yao, Z. Tang, and C. Qin, "Effective reversible data hiding in encrypted image with adaptive encoding strategy," *Information Sciences*, vol. 494, pp. 21–36, 2019.

[124] Z. Q. H. Ge, Y. Chen and J. Wang, "A high capacity multi-level approach for reversible data hiding in encrypted images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 8, pp. 2285–2295, 2019.

[125] S. Yi and Y. Zhou, "Separable and reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Transactions on Multimedia*, vol. 21, no. 1, pp. 51–64, 2019.

[126] P. C. M. Liu Z. L., "Reversible image reconstruction for reversible data hiding in encrypted images," *Signal Processing*, vol. 161, pp. 50–62, 2019.

[127] Z. Liu and C. Pun, "Reversible data hiding in encrypted images using chunk encryption and redundancy matrix representation," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2020.

[128] L. Luo, Z. Chen, M. Chen, X. Zeng, and Z. Xiong, "Reversible image watermarking using interpolation technique," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 1, pp. 187–193, 2010.

[129] X. Cao, L. Du, X. Wei, D. Meng, and X. Guo, "High capacity reversible data hiding in encrypted images by patch-level sparse representation," *IEEE Transactions on Cybernetics*, vol. 46, no. 5, pp. 1132–1143, 2016.

[130] W. Stallings, “Cryptography and network security: Principles and practice,” *3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2003.*, vol. 3, 1987.

[131] K. Chen and C. Chang, “High-capacity reversible data hiding in encrypted images based on extended run-length coding and block-based msb plane rearrangement,” *Journal of Visual Communication and Image Representations*, vol. 58, pp. 334–344, 2019.

[132] A. Mohammadi, M. Nakhkash, and M. A. Akhaee, “A high-capacity reversible data hiding in encrypted images employing local difference predictor,” *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2020.

[133] P. Puteaux and W. Puech, “An efficient msb prediction-based method for high-capacity reversible data hiding in encrypted images,” *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 7, pp. 1670–1681, 2018.

[134] Z. Yin, Y. Xiang, and X. Zhang, “Reversible data hiding in encrypted images based on multi-msb prediction and huffman coding,” *IEEE Transactions on Multimedia*, vol. 22, no. 4, pp. 874–884, 2020.

[135] A. Malik, H. X. Wang, Y. Chen, and A. N. Khan, “A reversible data hiding in encrypted image based on prediction-error estimation and location map,” *Multimedia Tools and Applications*, vol. 79, pp. 11 591–11 614, 2020.

[136] D. Huang and J. Wang, “High-capacity reversible data hiding in encrypted image based on specific encryption process,” *Signal Processing: Image Communication*, vol. 80, 2020.

[137] Z. Yin, B. Luo, and W. Hong, “Separable and error-free reversible data hiding in

encrypted image with high payload," *IEEE Trans. on Inf. Forensics and Security*, vol. 2014, pp. 1–1, 2020.

[138] Y. Qiu, Z. Qian, and L. Yu, "Adaptive reversible data hiding by extending the generalized integer transformation," *IEEE Signal Processing Letters*, vol. 23, no. 1, pp. 130–134, 2016.

[139] F. Peng, X. Li, , and B. Yang, "Adaptive reversible data hiding scheme based on integer transform," *Signal Processing*, vol. 92, no. 1, pp. 54–62, 2012.

[140] J. Zhou, W. Sun, L. Dong, X. Liu, O. C. Au, and Y. Y. Tang, "Secure reversible image data hiding over encrypted domain via key modulation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 3, pp. 441–452, 2016.

[141] X. Zhang, J. Long, Z. Wang, and H. Cheng, "Lossless and reversible data hiding in encrypted images with public-key cryptography," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1622–1631, 2016.

[142] P. Puteaux and W. Puech, "Epe-based huge-capacity reversible data hiding in encrypted images," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, 2018, pp. 1–7.

[143] P.Puteaux and W.Puech, "A recursive reversible data hiding in encrypted images method with a very high payload," *IEEE Transactions on Multimedia*, pp. 1–1, 2020.

[144] I. C. Dragoi and D. Coltuc, "On the security of reversible data hiding in encrypted images by msb prediction," *IEEE Transactions on Information Forensics and Security*, pp. 1–1, 2020.

[145] Y. Wu, Y. Xiang, Y. Guo, J. Tang, and Z. Yin, "An improved reversible data hiding

in encrypted images using parametric binary tree labeling," *IEEE Transactions on Multimedia*, 2019.

[146] M. Y. Y. L. H. S. H. Yao and T. Qiao, "Adaptive and separable multiary reversible data hiding in encryption domain," *EURASIP Journal on Image and Video Processing*, vol. 16, 2020.

[147] Y. Wu, W. Ma, Y. Peng, R. Zhang, and Z. Yin, "Reversible data hiding in encrypted images based on bit plane compression of prediction error," *Scientific World J.*, pp. 1–1, 2020.

[148] X. Wang, C. Shao, X. Xu, and X. Niu, "Reversible data-hiding scheme for 2-d vector maps based on difference expansion," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 3, pp. 311–320, 2007.

[149] X. Zhang and S. Wang, "Efficient steganographic embedding by exploiting modification direction," *IEEE Communications Letters*, vol. 10, no. 11, pp. 781–783, 2006.

[150] C.-C. Chang, T. D. Kieu, and Y.-C. Chou, "Reversible data hiding scheme using two steganographic images," in *TENCON 2007 - 2007 IEEE Region 10 Conference*, 2007, pp. 1–4.

[151] C.-C. Chang, T.-C. Lu, G. Horng, Y.-H. Huang, and Y.-M. Hsu, "A high payload data embedding scheme using dual stego-images with reversibility," in *2013 9th International Conference on Information, Communications Signal Processing*, 2013, pp. 1–5.

[152] C. Lee and Y. Huang, "Reversible data hiding scheme based on dual stegano-images using orientation combinations," *Telecommunication Systems*, vol. 52, pp. 2237–2247, 2013.

[153] C. Qin, C. Chang, and T. Hsu, “Reversible data hiding scheme based on exploiting modification direction with two steganographic images,” *Multimed Tools Appl*, vol. 74, pp. 5861–5872, 2015.

[154] J. Lin, Y. Chen, Chang, and C. et al., “Dual-image-based reversible data hiding scheme with integrity verification using exploiting modification direction,” *Multimed Tools Appl*, vol. 78, pp. 25 855–25 872, 2019.

[155] X. Chen and C. Hong, “An efficient dual-image reversible data hiding scheme based on exploiting modification direction,” *Journal of Information Security and Applications*, vol. 58, p. 102702, 2021.

[156] Y. J. Liu and C. C. Chang, “A turtle shell-based visual secret sharing scheme with reversibility and authentication,” 2018.

[157] C. C. Chang, Y. Liu, and T. S. Nguyen, “A novel turtle shell based scheme for data hiding,” in *2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, 2014, pp. 89–93.

[158] X.-Z. Xie, C.-C. Lin, and C.-C. Chang, “Data hiding based on a two-layer turtle shell matrix,” 2018.

[159] J. Lin, Y. Liu, and C. Chang, “A real-time dual-image-based reversible data hiding scheme using turtle shells,” *J Real-Time Image Proc*, vol. 16, pp. 673–684, 2019.

[160] X. Xie and C. Chang, “Hiding data in dual images based on turtle shell matrix with high embedding capacity and reversibility,” 2021.

[161] T.-C. Lu, J.-H. Wu, and C.-C. Huang, “Dual-image-based reversible data hiding method using center folding strategy,” *Signal Processing*, vol. 115, pp. 195–213, 2015.

[162] H. Yao, C. Qin, Z. Tang, and Y. Tian, “Improved dual-image reversible data hiding method using the selection strategy of shiftable pixels’ coordinates with minimum distortion,” *Signal Processing*, vol. 135, pp. 26–35, 2017.

[163] L. Chi, C. Wu, and H. Chang, “Reversible data hiding in dual stegano-image using an improved center folding strategy,” 2018.

[164] J. Biswapati, G. Debasis, and M. Kumar, “Dual-image based reversible data hiding scheme through pixel value differencing with exploiting modification direction,” in *First International Conference on Intelligent Computing and Communication*, vol. 458, 2017, pp. 89–93.

[165] P.-H. Kim, K.-W. Ryu, and K.-H. Jung, “Reversible data hiding scheme based on pixel-value differencing in dual images,” *International Journal of Distributed Sensor Networks*, vol. 16, no. 7, p. 1550147720911006, 2020.

[166] H. Yao, C. Qin, Z. Tang, and Y. Tian, “Improved dual-image reversible data hiding method using the selection strategy of shiftable pixels’ coordinates with minimum distortion,” *Signal Processing*, vol. 135, pp. 26–35, 2017.

[167] S. Shastri and V. Thanikaiselvan, “Dual image reversible data hiding using trinary assignment and centre folding strategy with low distortion,” *Journal of Visual Communication and Image Representation*, vol. 61, pp. 130–140, 2019.

[168] A. K. Sahu and G. Swain, “High fidelity based reversible data hiding using modified lsb matching and pixel difference,” *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 4, pp. 1395–1409, 2022.

[169] T.-C. Lu, C.-Y. Tseng, and J.-H. Wu, “Dual imaging-based reversible hiding technique using lsb matching,” *Signal Processing*, vol. 108, pp. 77–89, 2015.

[170] Z. Qian, H. Xu, X. Luo, and X. Zhang, “New framework of reversible data hiding in encrypted jpeg bitstreams,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 2, pp. 351–362, 2019.

[171] F. Chen, Y. Yuan, H. He, M. Tian, and H. Tai, “Multi-msb compression based reversible data hiding scheme in encrypted images,” *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2020.

[172] H. Jegou, M. Douze, and C. Schmid, “Hamming embedding and weak geometry consistency for large scale image search,” in *Proceedings of the 10th European conference on Computer vision, October, 2008*, 2008.

[173] J.A.Garcia and R. Sanchez, “Cvg-ugr image data base,” *http://decsai.ugr.es/cvg/dbimagenes /index.php*, 2002.

[174] G.Schaefer and M. Stich, “Ucid-an uncompressed color image database,” in *Proc. SPIE, Storage and Retrieval Methods and Applications for Multimedia*, vol. 472-480, 2004.

[175] Patrick and Teddy, “Bows-2,” in *Bows-2: Break over watermark system, Second Eddition*, 2008.

[176] D. Tao, “The corel database for content based image retrieval,” *xyz*, 2009.

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

[178] M. Xiao, X. Li, W. Lu, and Y. Zhao, “Histogram shifting based reversible data hiding with multiple expansion bin pairs,” *Displays*, vol. 83, p. 102674, 2024.

[179] Q. Ying, Z. Qian, X. Zhang, and D. Ye, “Reversible data hiding with image enhancement using histogram shifting,” *IEEE Access*, vol. 7, pp. 46 506–46 521, 2019.

[180] Z. Bian, H. Yao, Y. Le, and C. Qin, “Two-dimensional histogram-based reversible contrast enhancement using bi-histogram equalization,” *Displays*, vol. 81, p. 102580, 2024.

[181] Q. Chang, X. Li, and Y. Zhao, “Reversible data hiding for color images based on adaptive three-dimensional histogram modification,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 9, pp. 5725–5735, 2022.

[182] S. Kumar, G. S. Walia, and A. Gupta, “A multistage high-capacity reversible data hiding technique without overhead communication,” *Defence Science Journal*, vol. 73, no. 6, pp. 688–698, 2023.

[183] G. A. . W. G. Kumar S., “Reversible data hiding: A contemporary survey of state-of-the-art, opportunities and challenges,” *Applied Intelligence*, vol. 52, no. 7, pp. 7373–7406, 2022.

[184] J. F. D. C. De Vleeschouwer and B. Macq, “Circular interpretation on histogram for reversible watermarking,” *IEEE Int. Multimedia Signal Process. Workshop, France*, pp. 345–350, 2001.

[185] P. Garg, S. S. Kasana, and G. Kasana, “Reversible data hiding techniques with high message embedding capacity in images,” *Journal of Circuits, Systems and Computers*, vol. 26, no. 06, p. 1750103, 2017.

[186] F. Aziz, T. Ahmad, A. Malik, M. Uddin, S. Ahmad, and M. Sharaf, “Block-based reversible data hiding using histogram shifting and modulus operator for digital images,” *PLoS ONE*, vol. 5, no. 15, pp. 1–24, 2020.

[187] A. Malik, G. Sikka, and H. Verma, “Image interpolation based high capacity reversible data hiding scheme,” *Multimed Tools Appl.*, vol. 76, pp. 24 107–24 123, 2017.

[188] D. Xiao, Y. Wang, T. Xiang, and S.Bai, “High-payload completely reversible data hiding in encrypted images by an interpolation technique,” *Frontiers Inf. Technol. Electron. Eng*, vol. 18, pp. 1732–1743, 2017.

[189] A. Malik, A. Wang, T. Chen, T. Yang, A. Khan, H. Wu, Y. Chen, and Y. Hu, “Reversible data hiding in homomorphically encrypted image using interpolation technique,” *J. Inf. Secur. Appl.*, vol. 48, p. Art. no. 102374, 2019.

[190] R. Punia, A. Malik, and S. Singh, “An interpolation-based reversible data hiding scheme for internet of things applications,” *Discov Internet Things*, vol. 3, no. 18, 2023.

[191] X. Xiong , S. Zhong, and Y. Lu, “Separable and reversible data hiding scheme for medical images using modified logistic and interpolation,” *Biomedical Signal Processing and Control*, vol. 87, p. 105521, 2024.

[192] Y. Chang, C. Huang, C. Lee, and S. Wang, “Image interpolating based data hiding in conjunction with pixel-shifting of histogram,” *Journal of Super computing*, vol. 66, pp. 1093–1110, 2013.

[193] A. Malik, G. Sikka, and H. Verma, “A reversible data hiding scheme for interpolated images based on pixel intensity range,” *Multimed Tools Appl*, vol. 79, pp. 18 005–18 031, 2020.

[194] F. Hassan and A. Gutub, “An efficient reversible data hiding multimedia technique based on smart image interpolation,” *Multimed Tools Appl*, vol. 79, pp. 30 087–30 109, 2020.

[195] S. Zhong and Y. L. X. Xiong, “Reversible data hiding algorithm in encrypted domain based on image interpolation,” *IEEE Access*, vol. 11, pp. 108 281–108 294, 2023.

[196] X. Bai, Y. Chen, G. Duan, C. Feng, and W. Zhang, “A data hiding scheme based on the difference of image interpolation algorithms,” *Journal of Information Security and Applications*, vol. 65, 2022.

Biodata

Sanjay Kumar completed his M.Sc. (First class) degree in Mathematics at **Chaudhary Charan Singh University Campus (formerly Meerut University)**, Meerut, in 2005. Presently, he holds the position of Scientist 'E' in Scientific Analysis Group, DRDO, Ministry of Defence, Government of India. He has worked in the field of Symmetric Key Cryptography, Elliptic Curve Cryptography, and Post Quantum Cryptography. He joined Delhi Technological University, New Delhi as a part time Ph.D. Scholar in the Department of Applied Mathematics under the supervision of **Prof. Anjana Gupta** and **Dr. Gurjit Singh Walia** in 2018. His current research focuses on Reversible Data Hiding Techniques. He has proposed various robust and efficient methods in this research area.