REVERSIBLE DATA HIDING WITH IMAGE CONTRAST ENHANCEMENT BASED ON TWO DIMENSIONAL HISTOGRAM MODIFICATION

PROJECT WORK

PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF **BACHELOR OF TECHNOLOGY** IN **INFORMATION TECHNOLOGY** OF THE ANNA UNIVERSITY

Submitted by

KIRUTHIKA K
(1618117)
MANIMEGALAI N
(1618123)
MARLYN VICTORIA H
(1618124)

Under the Guidance of

Prof.T.Suguna M.Tech

2020

DEPARTMENT OF INFORMATION TECHNOLOGY
GOVERNMENT COLLEGE OF TECHNOLOGY
(An Autonomous Institution Affiliated to Anna University)
COIMBATORE - 641 013

DEPARTMENT OF INFORMATION TECHNOLOGY
GOVERNMENT COLLEGE OF TECHNOLOGY
(An Autonomous Institution affiliated to Anna University)
COIMBATORE - 641 013

PROJECT WORK

AUGUST 2020

This is to certify that this project work entitled

REVERSIBLE DATA HIDING WITH IMAGE CONTRAST ENHANCEMENT BASED ON TWO DIMENSIONAL HISTOGRAM MODIFICATION

is the bonafide record of project work done by

KIRUTHIKA K
(1618117)
MANIMEGALAI N
(1618123)
MARLYN VICTORIA H
(1618124)

of B.TECH. (Information Technology) during the year 2019 - 2020

__________________________ __________________________

Project Guide	Head of the Department
Prof.T.Suguna M.Tech	**Dr.S.Rathi M.E, Ph.D**

Submitted for the Project Viva-Voce examination held on ______________

Internal Examiner	External Examiner

ACKNOWLEDGEMENT

Great achievements are not possible without standing on the shoulders of giants. Without the active involvement of the following experts this project would not have been a reality.

We express our sincere gratitude to **Dr.P.Thamarai M.E, Ph.D.,** Principal, Government College of Technology, Coimbatore for providing us all facilities that we needed for the completion of this project.

We whole-heartedly express our thankfulness and gratitude to **Dr.S.Rathi M.E., Ph.D.,** Associate Professor and Head of the Department of Information Technology, Government College of Technology, for helping us to successfully carry out this project.

Our thankfulness and gratitude to our respectable project guide **Mrs.T.Suguna M.Tech.,** Assistant Professor, who has been immense help through the various phases of the project. With her potent ideas and excellent guidance, we were able to comprehend the essential aspects involved.

We would like to thank our faculty advisor **Mrs.C.Aswini M.E.,** Assistant Professor for her continuous support and encouragement throughout this project.

We extend our sincere thanks to the staff members of Information Technology department, **Mrs.R.Devi M.Tech.,** Assistant Professor, **Thiru.S.Gladson Oliver M.Tech.,** Assistant Professor, **Ms.M.Jeyanthi M.Tech.,** Assistant Professor, **Thiru.J.Kalidass M.E.,** Assistant Professor , **Mrs.R.Malavika M.Tech.,** Assistant Professor, **Dr.M.Blessy Queen Mary M.E.,Ph.D.,** Assistant Professor, for rendering their help for the completion of this project. We also thank all our friends for their cooperation and suggestions towards the successful completion of this project.

SYNOPSIS

Data hiding is an art, which involves the technique of carrying a data in a suitable multimedia carrier for secure communication. Data hiding techniques provide secret communication and authentication but can cause a loss of the carrier. These techniques are used for copyright protection, media registration, integrity authentication etc. Applications like medical imagery, military and forensics degradation do not allow distortion of original cover. So it needs secure data hiding techniques. To overcome this disadvantage of extracting the carrier with distortion was removed by reversible data hiding methods. Reversible data hiding techniques recovers the original carrier exactly after the extraction of the secret encrypted data. This technique ensures security and provides mechanism to protect the integrity of the message from any modification by preventing planned and accidental changes. The proposed method is based on 2D histogram modification to exploit the redundancy in digital images and to produce extra space for embedding to achieve very high embedding capacity.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATIONS

RDH	REVERSIBLE DATA HIDING
LSB	LEAST SIGNIFICANT ALGORITHM
CE	CONTRAST ENHANCEMENT
SSIM	STRUCTURAL SIMILARITY INDEX
MSE	MEAN SQUARE ERROR
PSNR	PEAK SIGNAL-TO- NOISE RATIO

CHAPTER 1
INTRODUCTION

1.1 DESCRIPTION

The availability of internet everywhere and the risks of illegal accessing of transmitted data is increasing day by day. Therefore, there is a need for protection of secret data from unauthorized users in a public network. Data hiding is one of the most important techniques to protect the security of digital media. Data hiding is used in a wide range of applications for embedding confidential messages. Sometimes hiding the information demolishes the host image even though the distortion is not visible to human eyes. In some applications such as medical imagery, military imagery and law forensics, no degradation of the original cover is allowed. In these cases, we need an important data hiding method known as Reversible Data Hiding or lossless data hiding. Reversible data hiding techniques are devised to solve the problems of lossless embedding of larger messages. The image can be restored completely without any loss. Contrast enhancement is executed to bring out unclear details and is widely used in medical images.

1.1.1 REVERSIBLE DATA HIDING ALGORITHM

Reversible Data Hiding (RDH) has been intensively studied in the community of signal processing. Also referred to as invertible or lossless data hiding, RDH is used to embed a piece of information into a host signal to generate the marked one, from which the original signal can be exactly recovered after extracting the embedded data. The technique of RDH is useful in some sensitive applications where no permanent change is allowed on the host signal. Most of the proposed algorithms are for digital images to embed invisible data or a visible watermark. To evaluate the performance of a RDH algorithm, the hiding rate and the marked image quality are important metrics. There exists a trade-off between them because increasing the hiding rate often causes more distortion in image content.

1.2 EXISTING SYSTEM

In the existing methods, some of the disadvantages are listed.

- Recovery of secret message may fail due to pixel similarities.
- Auxillary information needs to be provided for data extraction.
- It is difficult to distinguish prediction error.
- Both encryption key and data hiding key required to extract data.
- Low embedding capacity & unconditional security.

1.3 PROBLEM DEFINITION

- This project attempts to make use of two dimensional histogram to perform the task of data hiding for the purpose of preserving privacy.
- Though many data hiding methods do the same, our algorithm provides contrast enhancement of images & high data embedding rate.
- Data hiding in reversible manner will provide double security for the data such as image encryption as well as data hiding in encrypted images.

1.4 PROPOSED SYSTEM

- A reversible data hiding algorithm that hides a bit stream of the secret text in to the least significant bits(LSBs) of the approximation coefficients of each pixel of color images to form contrast-enhanced images is proposed.
- The embedding and extracting phases of the proposed reversible data hiding algorithms are performed using the MATLAB software. Invisibility, payload capacity, and security in terms of peak signal to noise ratio (PSNR) and robustness are the key challenges.
- The statistical distortion between the cover images and the contrast-enhanced images is measured by using the mean square error (MSE) and the PSNR.

1.5 ORGANIZATION OF THE PROJECT

- Literature reviews of already existing proposals are discussed in chapter 2.
- Chapter 3 has system specification which tells about the software and hardware requirements.
- Chapter 4 discusses the overall project and design which tells the brief description of each of the modulus in this project.
- Chapter 5 has the implementation and experimental result of the project.
- Chapter 6 deals with the conclusion and future work.
- Finally chapter 7 deals with the references.

CHAPTER 2

LITERATURE REVIEW

2.1 NEW FRAMEWORK OF REVERSIBLE DATA HIDING IN ENCRYPTED JPEG BITSTREAMS [2019]

2.1.1 DESCRIPTION

This paper proposes a novel framework of reversible data hiding in encrypted JPEG bitstream. We first provide a JPEG encryption algorithm to encipher a JPEG image to a smaller size and keep the format compliant to JPEG decoders. After an image owner uploads the encrypted JPEG bitstreams to cloud storage, the server embeds additional messages into the ciphertext to construct a marked encrypted JPEG bitstream. During data hiding, we propose a combined embedding algorithm including two stages, the Huffman code mapping and the ordered histogram shifting. The embedding procedure is reversible. When an authorized user requires a downloading operation, the server extracts additional messages from the marked encrypted JPEG bitstream and recovers the original encrypted bitstream losslessly. After downloading, the user obtains the original JPEG bitstream by a direct decryption. The proposed framework outperforms previous works on reversible data hiding in encrypted images. First, since the tasks of data embedding/extraction and bitstream recovery are all accomplished by the server, the image owner and the authorized user are required to implement no extra operations except JPEG encryption or decryption. Second, the embedding payload is larger than state-of-the-art works.

2.1.2 MERIT

- Embedding payload is larger.

2.1.3 DEMERIT

- When pixels have identical values, data extraction and image recovery may fail.

2.2 HIGH CAPACITY REVERSIBLE DATA HIDING IN ENCRYPTED IMAGES BY BIT PLANE PARTITION AND MSB PREDICTION [2019]

2.2.1 DESCRIPTION

This paper proposes a high-capacity scheme for reversible data hiding (RDH) in encrypted images. The proposed scheme is based on a new preprocessing method by bit plane partition. Specifically, the values in the less significant bit planes are reversibly hidden into the other bit planes. Consequently, the room in the less significant bit planes can be vacated to generate a preprocessed image, from which the original image can be recovered by extracting the hidden bit values and writing them back. After encrypting the preprocessed image with a stream cipher, the vacated room can be used to accommodate extra data, which can be retrieved without image decryption. In addition, the embedding capacity can be further increased by adopting an efficient most significant bit (MSB) prediction method before image encryption. Compared with the state-of-the-art RDH schemes for encrypted images, the experimental results show that higher embedding capacity can be achieved with our proposed one. The numerical results are provided to show the performances of the proposed scheme in different cases of bit-plane partition and MSB prediction.

2.2.2 MERIT

- Embedding capacity can be increased.

2.2.3 DEMERIT

- Auxiliary information need to be separately provided for data extraction.

2.3 REVERSIBLE DATA HIDING IN ENCRYPTED IMAGES BASED ON MULTI-MSB PREDICTION AND HUFFMAN CODING [2019]

2.3.1 DESCRIPTION

With the development of cloud storage and privacy protection, reversible data hiding in encrypted images (RDHEI) has attracted increasing attention as a technology that can: embed additional data in the image encryption domain, ensure that the embedded data can be extracted error-free, and the original image can be restored losslessly. In this paper, a high-capacity RDHEI algorithm based on multi-MSB (most significant bit) prediction and Huffman coding is proposed. At first, multi-MSB of each pixel was predicted adaptively and marked by Huffman coding in the original image. Then, the image was encrypted by a stream cipher method. At last, the vacated space can be used to embed additional data by multi-MSB substitution. Experimental results show that our method achieved higher embedding capacity while comparing with the state-of-the-art methods.

2.3.2 MERIT

- Embedded data can be error free and Higher embedding capacity.

2.3.3 DEMERIT

- Distinguishing prediction errors is difficult.

2.4 A NEW REVERSIBLE DATA HIDING IN ENCRYPTED IMAGE ON MULTISECRET SHARING AND LIGHTWEIGHT CRYPTOGRAPHIC ALGORITHM [2019]

2.4.1 DESCRIPTION

Reversible data hiding in encrypted images (RDHEI) has been introduced for preserving image privacy and data embedding. RDHEI usually involves three parties, namely, the image provider, data hider, and receiver. On the security with key setting, there are three categories: share independent secret keys (SIK), shared one key (SOK), and share no secret keys (SNK). In SIK, the image provider and data hider must respectively and independently share secret keys with the receiver, whereas in SNK, no secret key is shared. However, the literature works proposed SNK-type schemes by using homomorphic encryption (with exorbitant computation cost). In this paper, we address the SOK setting, where only the image provider shares a secret key with the receiver, and the data hider can embed a secret message without any knowledge of this key. To realize our SOK scheme in a simple manner, we propose a new technique by using multi-secret sharing as the underlying encryption, which indeed induces a blow-up issue of the key size. For preserving the efficiency of the key size, we apply a compression by using lightweight cryptographic algorithms. Then, we demonstrate our SOK scheme based on the proposed techniques, and show effectiveness, efficiency, and security by experiments and analysis.

2.4.2 MERIT

- Improved payload and image quality.

2.4.3 DEMERIT

- Unconditional security.

2.5 REAL-TIME REVERSIBLE DATA HIDING IN ENCRYPTED IMAGES BASED ON HYBRID EMBEDDING MECHANISM [2018]

2.5.1 DESCRIPTION

In this paper, we propose a novel real-time scheme of separable reversible data hiding in encrypted images, which consists of image encryption, data embedding, data extraction and image recovery. In image encryption phase, the content owner divides the original image into a number of non-overlapping blocks and encrypts blocks by stream cipher and permutation .During the data embedding phase, the data hider classifies encrypted blocks into smooth region and complex region according to the threshold and replaces the MSB layer of a part of pixels in blocks of smooth region with the secret data. Then, the LSB layers of other pixels are collected and compressed to generate a room for embedding the secret data again. When the receiver receives the marked image, he can divide the marked image into blocks and decrypt them by the encryption key to obtain a similar image with good quality. If the receiver only has the data hiding key, he can classify the blocks into smooth region and complex region according to the threshold and extract the embedded data by the data hiding key. If the receiver has both encryption key and data hiding key, he can extract the embedded data from the marked image and recover the original image perfectly. The proposed scheme can achieve satisfactory quality of decrypted image and high embedding rate. Experimental results demonstrate the effectiveness and computational efficiency of our scheme.

2.5.2 MERIT

- Satisfactory quality of decrypted image and high embedding rate.

2.5.3 DEMERIT

- Both encryption key and data hiding key required to extract data.

2.6 AN EFFICIENT REVERSIBLE DATA HIDING SCHEME BASED REFERENCE PIXEL AND BLOCK SELECTION [2013]

2.6.1 DESCRIPTION

In this paper, we propose an efficient data hiding scheme based on reference pixel and block selection to further improve the embedding performance of histogram shifting. Specifically, we first divide the original image into non-overlapping blocks of an adjustable size. Then for each block, we assign the median of pixels as the reference pixel and the number of pixels equal to the reference value as the smooth level. In this way, difference histograms for each smooth level can be constructed. We embed the secret data using histogram shifting from the highest level histogram to lower level ones instead of sequential embedding. By this means, our proposed reversible data hiding scheme can adaptively embed data in the smooth blocks and thus improve the marked image quality with a comparable embedding capacity. The experimental results also demonstrate its superiority over some state-of-the-art reversible data hiding works.

2.6.2 MERIT

- Improve the marked image quality.

2.6.3 DEMERIT

- Low embedding capacity.

2.7 A NOVEL APPROACH TOWARDS SEPARABLE REVERSIBLE DATA HIDING TECHNIQUE [2014]

2.7.1 DESCRIPTION

Internet is the most popular communication medium now a days but message communication over the internet is facing some problem such as data security, copyright control, data size capacity, authentication etc. There are so many research is progressing on the field like internet security, steganography, cryptography. When it is desired to send the confidential/important/secure data over an insecure and bandwidth-constrained channel it is customary to encrypt as well as compress the cover data and then embed the confidential/important/secure data into that cover data. This paper introduces the new way of originating the existing concept i.e. separable reversible data hiding. Actually the concept of separable reversible data hiding technique is based on steganography and related with internet security. The chief objectives of this literature is to work on the concept in which we used text as a hidden data, no plain spatial domain is used, attempt to increase the amount of data which is to be hide, evaluating quality by different interpretations. The principal notion of separable reversible data hiding is consist of three key procedures. First encrypt the cover media second hide the data and third get the data as well as cover media as per provisions.

2.7.2 MERIT

- Attempt to increase the amount of data which is to be hide.

2.7.3 DEMERIT

- Security is less.

2.8 REVERSIBLE DATA HIDING WITH AUTOMATIC BRIGHTNESS PRESERVING CONTRAST ENHANCEMENT [2018]

2.8.1 DESCRIPTION

Reversible data hiding with automatic contrast enhancement methods provide an interoperable way to reduce the storage requirement for automatic image enhancement applications: original image can be recovered from the enhanced image without any additional information. Unlike the previous work, where the goal was to maximize the contrast, the proposed method increases the contrast to an appropriate level using an idea called brightness preservation. This is achieved by using an adaptive bin selection process based on the original brightness. Extensive experimental results verify that the enhanced images produced using the proposed method are visually and quantitatively superior than the existing work.

2.8.2 MERIT

- Reduce the storage requirement.

2.8.3 DEMERIT

- Image quality is not good

2.9 HISTOGRAM SHIFTING BASED REVERSIBLE DATA HIDING WITH CONTROLLED CONTRAST ENHANCEMENT [2017]

2.9.1 DESCRIPTION

This paper presents an improved reversible data hiding algorithm using digital images based on the histogram shifting technique. Proposed method can accurately recover the original image and extract the hidden data accurately. The highest two peak values of the host image's histogram are selected for data hiding. This embedding process is repeated again and again, to attain larger embedding capacity. In addition, a Controlled Contrast Enhancement (CCE) is performed to get good visual perception. To verify the robustness of the proposed method various attacks on the host image such as impulse noise, shearing, rotation, and scaling are considered. The proposed algorithm efficiently removes the various attacks on the watermarked image and efficiently recovers the original and embedded data. The proposed method's better performance over existing methods in terms of PSNR, SSIM and embedding rate is demonstrated.

2.9.2 MERIT

- Efficiently removes the various attacks on the watermarked image.

2.9.3 DEMERIT

- It satisfies low payload capacity.

2.10 REVERSIBLE DATA HIDING WITH IMAGE BIT-PLANE SLICING [2017]

2.10.1 DESCRIPTION

Utilizing bit-plane slicing, a novel approach for reversible data hiding (RDH) is introduced in this paper. Instead of directly embedding in an input image, we propose to embed in a pair of bit-plane sliced images of the input image. Specifically, an (m + n)-bit input image is subdivided in two lower intensity images, i.e., n-bit image using n-LSB planes and m-bit image using m MSB planes. Embedding in a lower intensity image would offer relatively higher embedding rate, since the pixel-counts of the highest bin in the image histogram would be much higher than that of the original image. Moreover, embedding in the n-bit image would cause lower embedding distortion, while that in the m-bit image should contribute to a higher contrast enhancement. After embedding, histogram shifting (HS)-based embedding, those two images can be combined to get the (m+n)-bit embedded image. Comparing with a prominent HS-based RDH scheme, the proposed scheme has demonstrated significantly higher embedding rate and better contrast-enhancement.

2.10.2 MERIT

- Higher embedding rate.

2.10.3 DEMERIT

- Takes more execution time.

2.11 REVERSIBLE DATA HIDING WITH IMAGE ENHANCEMENT USING HISTOGRAM SHIFTING [2019]

2.11.1 DESCRIPTION

Traditional reversible data hiding (RDH) focuses on enlarging the embedding payloads while minimizing the distortion with a criterion of mean square error (MSE). Since imperceptibility can also be achieved via image processing, we propose a novel method of RDH with contrast enhancement (RDH-CE) using histogram shifting. Instead of minimizing the MSE, the proposed method generates marked images with good quality with the sense of structural similarity. The proposed method contains two parts: the baseline embedding and the extensive embedding. In the baseline part, we first merge the least significant bins to reserve spare bins and then embed additional data by a histogram shifting approach using arithmetic encoding. During histogram shifting, we propose to construct the transfer matrix by maximizing the entropy of the histogram. After embedding, the marked image containing additional data has a larger contrast than the original image. In the extensive embedding part, we further propose to concatenate the baseline embedding with an MSE-based embedding. On the recipient side, the additional data can be extracted exactly, and the original image can be recovered losslessly. Comparing with existing RDH-CE approaches, the proposed method can achieve a better embedding payload.

2.11.2 MERIT

- Generates marked images with good quality and better embedding payload.

2.11.3 DEMERIT

- It causes distortion.

2.12 REVERSIBLE DATA HIDING USING CONTROLLED CONTRAST ENHANCEMENT AND INTEGER WAVELET TRANSFORM [2015]

2.12.1 DESCRIPTION

The conventional reversible data hiding (RDH) algorithms pursue high Peak-Signal-to-Noise-Ratio (PSNR) at the certain amount of embedding bits. Recently, Wu et al. deemed that the improvement of image visual quality is more important than keeping high PSNR. Based on this viewpoint, they presented a novel RDH scheme, utilizing contrast enhancement to replace the PSNR. However, when a large number of bits are embedded, image contrast is over-enhanced, which introduces obvious distortion for human visual perception. Motivated by this issue, a new RDH scheme is proposed using the controlled contrast enhancement (CCE) and Haar integer wavelet transform (IWT). The proposed scheme has large embedding capacity while maintaining satisfactory visual perception. Experimental results have demonstrated the effectiveness of the proposed scheme.

2.12.2 MERIT

- Large embedding capacity.

2.12.3 DEMERIT

- They have high memory requirements.

2.13 REVERSIBLE IMAGE DATA HIDING WITH IMAGE CONTRAST ENHANCEMENT [2014]

2.13.1 DESCRIPTION

In this letter, a novel reversible data hiding (RDH) algorithm is proposed for digital images. Instead of trying to keep the PSNR value high, the proposed algorithm enhances the contrast of a host image to improve its visual quality. The highest two bins in the histogram are selected for data embedding so that histogram equalization can be performed by repeating the process. The side information is embedded along with the message bits into the host image so that the original image is completely recoverable. The proposed algorithm was implemented on two sets of images to demonstrate its efficiency. To our best knowledge, it is the first algorithm that achieves image contrast enhancement by RDH. Furthermore, the evaluation results show that the visual quality can be preserved after a considerable amount of message bits have been embedded into the contrast-enhanced images, even better than three specific MATLAB functions used for image contrast enhancement.

2.13.2 MERIT

- Visual quality can be preserved.

2.13.3 DEMERIT

- It does only simple calculations.

2.14 REVERSIBLE DATA HIDING IN MEDICAL IMAGE FOR CONTRAST ENHANCEMENT OF ROI [2016]

2.14.1 DESCRIPTION

Reversible data hiding is a kind of information hiding technique that can exactly recover the original image through data hiding and extraction. It can be potentially used in the medical and military applications. In the literature, by using the maximum inter-class square error to separate the background and foreground, the principal gray-scale values in the segmented background can be identified. By excluding the corresponding histogram bins from being expanded, the contrast of region of interest (ROI) in medical images can be selectively enhanced. In this paper, a new method is proposed to get the foreground contour to enhance the contrast of ROI in medical images, while the original image can be exactly recovered when needed. The experimental results on a set of medical images show that the visibility of ROI can be improved. Compared with the previous method, the proposed method can achieve better contrast enhancement effects and visual quality.

2.14.2 MERIT

- Better contrast enhancement effects and visual quality.

2.14.3 DEMERIT

- Information need to be separately provided for data extraction.

2.15 AUTOMATIC CONTRAST ENHANCEMENT USING REVERSIBLE DATA HIDING [2015]

2.15.1 DESCRIPTION

Automatic image enhancement is increasingly becoming a popular tool for the smart phone environment. The tool automatically enhances the image right after it has been stored to enhance user's experience. But, because enhancements are subjective and dependent on the image, the original image is backed up to provide a recovery option. This requirement inevitably increases the storage requirement. In order to reduce it, a novel automatic contrast enhancement based on reversible data hiding (ACERDH) is proposed. The proposed method mimics the equalization effect observed in the basic contrast enhancement technique called global histogram equalization, while providing reversibility. The experiment visually show improved contrast. Additional experiment was done to compare the embedding capacity with an another reversible data hiding based contrast enhancement technique [4]. The proposed method is fit for automation, while providing data hiding capability and removing the additional storage requirement.

2.15.2 MERIT

- Data hiding capability and removing the additional storage requirement.

2.15.3 DEMERIT

- Image recovery may fail.

CHAPTER 3

SYSTEM SPECIFICATION

3.1 SYSTEM REQUIREMENTS

3.1.1 HARDWARE REQUIREMENTS

- ➢ System : Pentium IV 2.4 GHz.
- ➢ Hard Disk : 1 TB
- ➢ Ram : 512 Mb.

3.1.2 SOFTWARE REQUIREMENTS

- ➢ Operating system : Windows 10.
- ➢ Coding Language : MATLAB
- ➢ Tool : MATLAB R 2014

3.2 SOFTWARE DESCRIPTION

3.2.1 ABOUT MATLAB

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include:

- Math and computation
- Algorithm development
- Modeling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C or Fortran.

The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects, which together represent the state-of-the-art in software for matrix computation.

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.

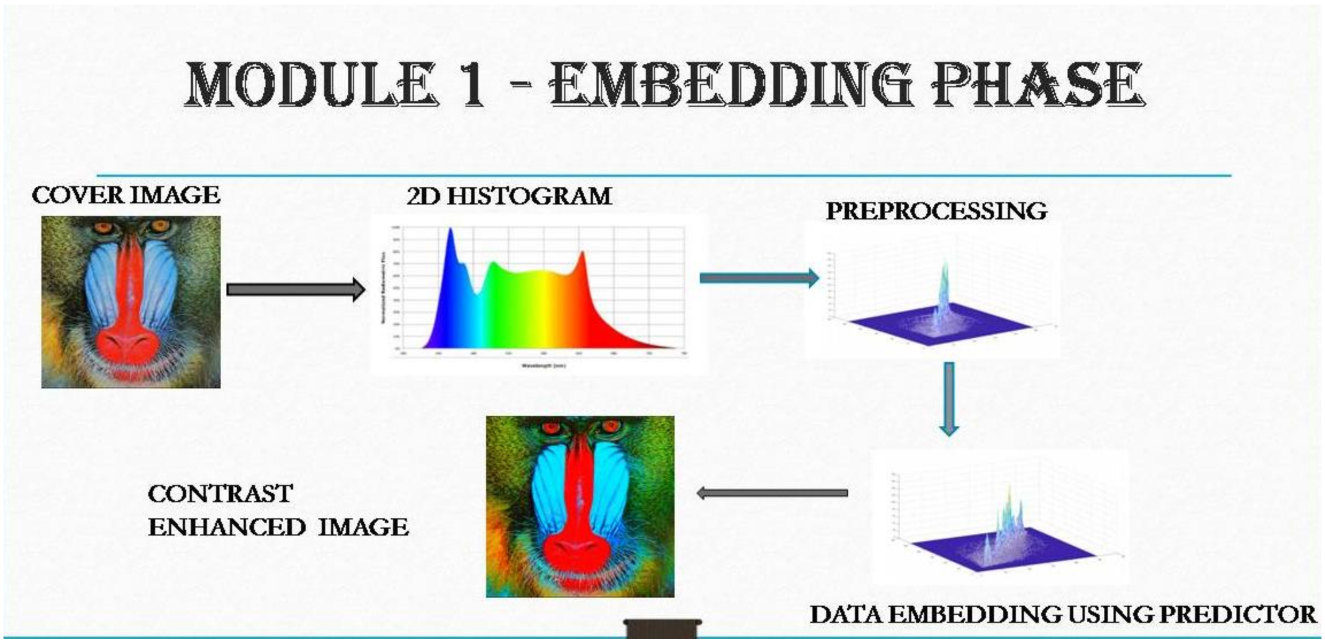
3.2.2 CHARACTERISTICS OF MATLAB

- High level language for scientific and engineering computing.
- Desktop environment tuned for iterative exploration, design and problem solving.
- Graphics for visualizing data and tools for creating plots.
- Apps for curve fitting, data classification, signal analysis and many other specific tasks.
- Add on tool boxes for a wide range of engineering and scientific applications.
- Tools for building applications with custom user interfaces.
- Interfaces to C/C++, Java, .NET, Python, SQL, Hadoop and Microsoft Excel.
- Royalty free deployment options for sharing MATLAB programs with end user.

CHAPTER 4

PROJECT DESIGN

4.1 MODULE DESCRIPTION OF EMBEDDING PHASE



4.1 FIGURE – EMBEDDING PHASE

Inputs: Cover image and secret text

Outputs: Embedded image (image containing a hidden secret text)

Step 1 Read the cover color image(I).

Step 2 Generate a 2D histogram of *I* and globally shifting the histogram.

Step 3 Perform preprocessing by merging adjacent rows and columns on each side for S times of a two dimensional histogram and generating the location map.

Step 4 Append to the payload certain information such as the location map's size, location map and Least Significant Bits of t+2 border pixels.

Step 5 Perform the steps (6 - 9) repeatedly for each predictor.

Step 6 Replace the t+2 border pixel Least Significant Bits with the payload length and predictor number.

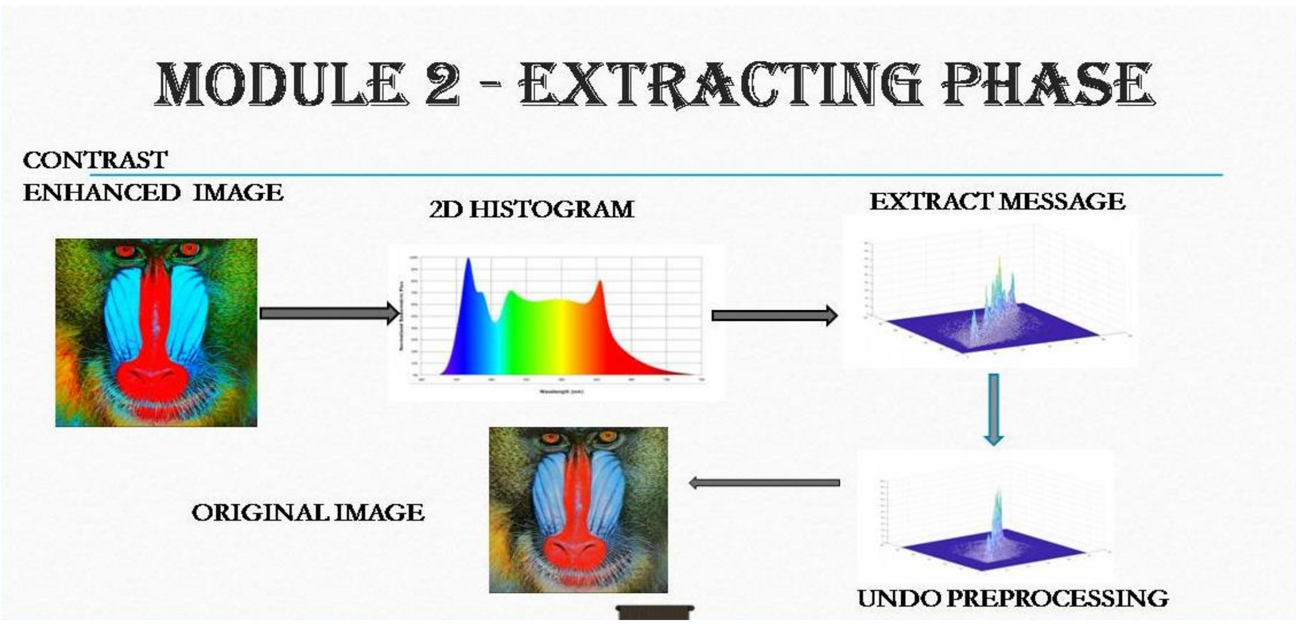
Step 7 Sort the even pixels. Use Positive Histogram Shifting and then Negative Histogram Shifting to embed the first half of payload.

Step 8 Sort the odd pixels. Use Positive Histogram Shifting and then Negative Histogram Shifting to embed the first half of payload.

Step 9 Calculate the PSNR between original and each of the embedded images.

Step 10 Generate the embedded image with the highest PSNR.

4.2 MODULE DESCRIPTION OF EXTRACTING PHASE



4.2 FIGURE- EXTRACTING PHASE

Inputs: Embedded image

Outputs: Original image, Secret text

Step 1 Read the embedded image.

Step 2 Collect the t+2 border pixel least significant bits and to extract the predictor number and payload length.

Step 3 Sort the odd pixels. Use Positive Histogram Shifting and then Negative Histogram Shifting to extract the second half of payload.

Step 4 Sort the even pixels. Use Positive Histogram Shifting and then Negative Histogram Shifting to extract the first half of payload.

Step 5 Replace the t+2 border pixel Least Significant Bits with the original least significant bits.

Step 6 Reverse the preprocessing strategy by uncompressing the location map.

Step 7 Use the extracted global shifting information to shift the whole 2D histogram to recover the original image.

Step8 The extracted image is now formed.

Step9 The secret text is now extracted.

4.3 PREPROCESSING

Preprocessing is done to prevent the overflow and underflow of pixel values due to histogram shifting. By embedding a pixel, we can get a watermarked pixel with overflow and underflow values resulting in 256 and -1.To solve this, we can preprocess the pixel values 0 and 255 to 1 and 254.The steps followed here consist of modifying the image before embedding and embed the location map to the payload. The location map is generated and original pixel p_o is preprocessed to preprocessing pixel in the following ways:

1) Scan the image and set the value to avoid overflow and underflow of pixel values.

$$M_i = \begin{cases} 0 & \text{if } p_o = 1 \text{ or } 254 \\ 1 & \text{if } p_o = 0 \text{ or } 255 \end{cases} \longrightarrow \text{Equation 1}$$

$$p = \begin{cases} 254 & \text{if } p_o=255 \\ 1 & \text{if } p_o=0 \\ p_0 & \text{else} \end{cases} \longrightarrow \text{Equation 2}$$

2) Repeat Equation 1 and 2, until the image is scanned. During the extraction stage,recovery of pixel values 0 and 255 using M is insignificant:

i.Scan the image and set the values.

$$p_o = \begin{cases} 0 & \text{if } p = 1 \text{ and } M_i = 1 \\ 255 & \text{if } p = 254 \text{ and } M_i = 1 \end{cases} \longrightarrow \text{Equation 3}$$

ii.Repeat the above steps until the whole image is scanned.

The location map is compressed losslessly using arithmetic coding and is added to the payload.

4.4 DATA EMBEDDING

Even sets are embedded first and then the odd sets. If the sum of the horizontal and vertical position is even, then the pixel is considered to be even, likewise odd if the sum is odd.

Each pixel is embedded twice in the same row, using two estimates namely, the high estimate (p_h) and low estimate (p_l).

The first prediction error histogram is $p\text{-}p_h$. One bit is embedded in p using Positive Histogram Shifting:

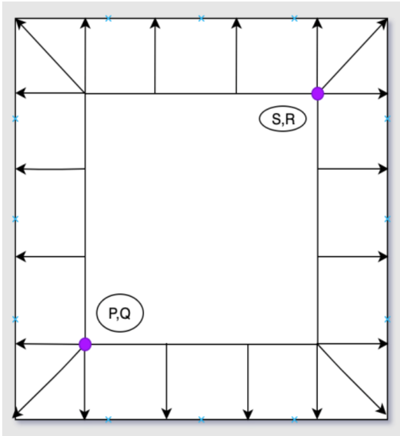
$$p' = \begin{cases} p + b_1 & \text{if } p -- p_h = 0 \\ p + 1 & \text{if } p - p_h > 0 \\ p & \text{else} \end{cases} \longrightarrow \text{Equation 4}$$

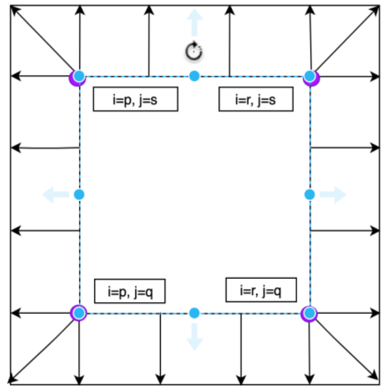
where $b_1 \in \{0,1\}$ is the first message bit and p' is the embedded pixel. If the prediction error is 0, only one bit is embedded and the rest of the pixels are shifted by 1 if prediction error is strictly positive.

The second prediction error histogram is $p'\text{-}p_l$. One bit is embedded in p' using Negative Histogram Shifting:

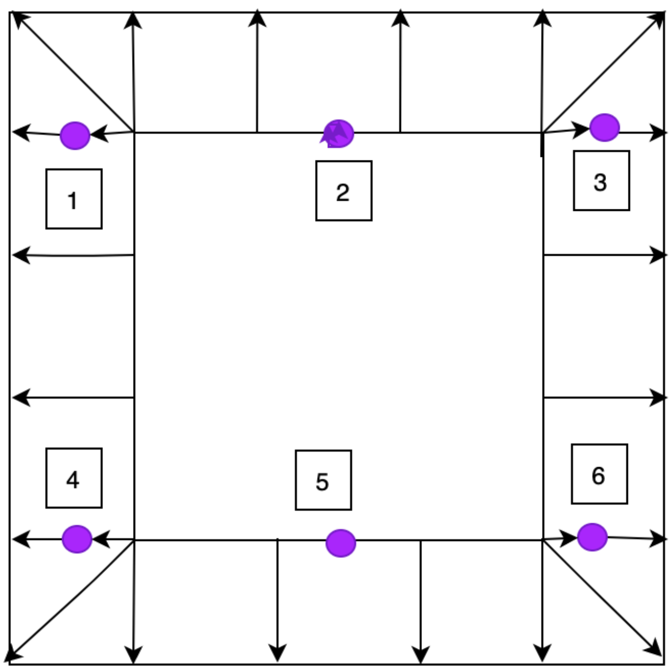
$$p'' = \begin{cases} p' - b_2 & \text{if } p' - p_l = 0 \\ p' - 1 & \text{if } p' - p_l < 0 \\ p' & \text{else} \end{cases} \longrightarrow \text{Equation 5}$$

where $b_2 \in \{0,1\}$ is the second message bit and p'' is the embedded pixel. If the prediction error is 0, only one bit is embedded and the rest of the pixels are shifted if prediction error is strictly negative.





(i + ba, j + bb), if i = r and j = s
(i – bc, j + bd), if i = p and j = s
(i – be, j – bf), if i = p and j = q
(i + bg, j – bh), if i = r and j = q



1. (i + bk , j – 1), if i = r and j < q
2. (i + bj, j), if i = r and q < j < s
3. (i + bi, j + 1), if i = r and j > s
4. (i – bn, j – 1), if i = p and j < q
5. (i – bm, j), if i = p and q < j < s
6. (i – bl, j + 1), if i = p and j > s

Fig 4.4.1. Embedding process in 2D histogram. This diagram depicts the data embedding process in 2D histogram using the above mentioned formula.

For eg:

p=50, q=50, r=200, s=200

(i,j) = (50,250)

i=50 => 10001100

j=250 => 11111010

Matched => i =p, j>s => (i-b,j+1)

Message bit (b) => 1

i-b=> 00110010

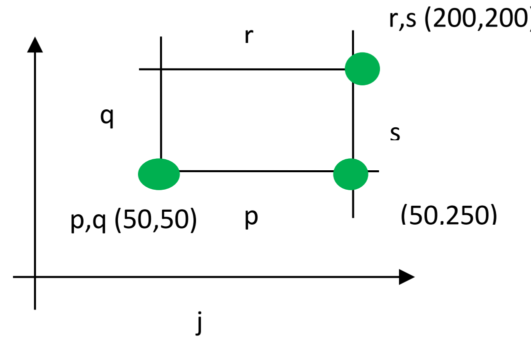
00000001

00110001

j+1 => 11111010

00000001

11111011



`

4.5 DATA EXTRACTION

The steps used in embedding are performed in reverse order for extraction and recovery, odd pixels are recovered first before even pixels. Message bits are extracted using following equation:

$$b_2 = \begin{cases} 0 & \text{if } p'' - p_l = 0 \\ 1 & \text{if } p'' - p_l = -1 \end{cases} \longrightarrow \text{Equation 6}$$

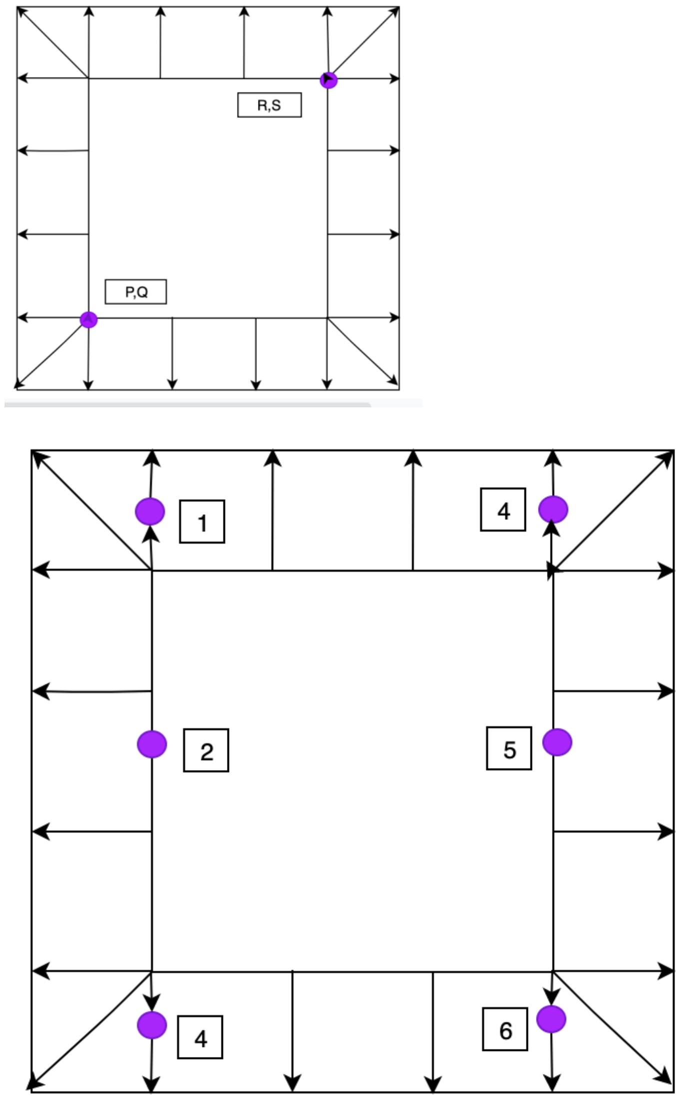
$$b_1 = \begin{cases} 0 & \text{if } p' - p_h = 0 \\ 1 & \text{if } p' - p_h = 1 \end{cases} \longrightarrow \text{Equation 7}$$

Pixels are recovered using the given equations:

$$p' = \begin{cases} p'' + 1 & \text{if } p'' - p_l < 0 \\ p'' & \text{else} \end{cases} \longrightarrow \text{Equation 8}$$

$$p = \begin{cases} p' - 1 & \text{if } p' - p_h > 0 \\ p' & \text{else} \end{cases} \longrightarrow \text{Equation 9}$$

The above mentioned steps are repeated for even set. The embedding capacity and the distortion will vary depending on different predictions.



1. (i + 1, j – br), if j = q and i > r
2. (i, j – bs), if j = q and p < i < r
3. (i – 1, j – bt), if j = q and i < p
4. (i + 1, j + bo), if j = s and i > r
5. (i, j + bp), if j = s and p < i < r
6. (i – 1, j + bq), if j = s and i < p

Fig 4.5.1. Extraction process in 2D histogram. This diagram depicts the data extraction process in 2D histogram using the above mentioned formula.

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1 IMPLEMENTATION

Implementation is the most crucial stage in achieving a successful system and giving the users confidence that new system is effective and workable. Implementation of this project refers to the installation of the packages in its real environment to the full satisfaction of the users and operations of the system. Testing is done individually at the time of development using the data and verification is done the way specified in the program specification. In short, implementation constitutes all activities that are required to put an already tested and completed package into operation. The success of any information system lies in its successful implementation. System implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and in giving confidence on the new system for the user that it will work efficiently and effectively. The existing system was long time process.

5.1.1 main.m

In the main function, embedding process is called first and after that extraction process will be carried out. After completion of the extraction phase MSE, PSNR and SSIM values will be calculated.

```
function main
img=imread('4.2.06.tiff');
original_image=double(rgb2gray(img));
[rows columns] = size(original_image);
if ( size (img,3 ) ~= 1)
   ig2=rgb2gray(img);
end

M='secret.txt';
secret=fopen(M,'rb');
original_message=reshape(dec2bin(secret,8).'-'0',1,[]);
fid = fopen('secret.txt');  % see also fclose
A = char(fread(fid,inf)).';
Z=reshape(dec2bin(A,8).'-'0',1,[]);
fprintf('Characters embedded:%d\n No.of bits embedded:%d\n',length(A),length(Z));
fprintf('\n');


embedding_Image= main_encode(original_image,original_message); %Perform encoding
```

```
img1=double(grs2rgb(embedding_Image)); %stego image to grayscale
if ( size (img1,3 ) ~= 1)
   ig3=rgb2gray(img1);
end

%Decoding check
[re_original, re_message]=main_decode(embedding_Image); %perform decoding

if isempty(embedding_Image)
   disp('failed to embed')
else
   disp(['Best psnr for the payload is ' num2str(psnr(original_image,embedding_Image,255))])
end

fprintf('\n');
%check if extracted messages are correct and pixels are recovered
if isequal(re_original,original_image)
   disp('Recovered the original image')
else
   disp('Failed to recover the original image')
end

if isequal(re_message,original_message)
   disp('Recovered the payload')
else
   disp('Failed to recover the payload')
end

fprintf('\n');
%Displaying the Images
figure,
subplot(1,2,1);
imshow(img);
title('Original Image');
imshow(img1);
title('Embedding Image');

%Displaying the Histograms
figure,
subplot(2,1,1);
imhist(ig2);
title('Histogram of Original Image');
subplot(2,1,2);
imhist(ig3);
title('Histogram of Embedded Image');


squaredErrorImage = (double(original_image) - double(embedding_Image)) .^ 2;
mse = sum(sum(squaredErrorImage)) / (rows * columns);
fprintf('MSE:%2f\n',mse);
b =ssim(original_image,embedding_Image);
fprintf('SSIM: %f\n',b);
```

Functions:

5.1.2 main_encode.m

This function is responsible for embedding data in the cover image and convert it into stego image. The whole encoding process involves preprocessing the image and embedding the secret text in the preprocessed image.

```
function [embedding_Image]=main_encode(original_image,original_message)
%Preprocess
[n m]=size(original_image);
message_length_max=floor(log2(n*m)); %less than or equal to n*m
length_loc_map_max=ceil(log2((n-2)*(m-2))); %nearest integer greater than or equal to n-2 *m-2

[P location_map location_map_length]=preproces_image(original_image); %P-image\
border_pixel=mod(P(1:message_length_max+2),2); % returns remainder of division
message=[border_pixel location_map_length location_map original_message];
total_message_length=length(message);
result=zeros(1,3);

%Encoding
for mode=1:3
   re_P=P;

   %Preprocess border pixels
   bi_message_length=de2bi(total_message_length,message_length_max); %converts non-negative
decimal integer to binary row vector

re_P(1:message_length_max+2)=bitxor(bitxor(P(1:message_length_max+2),border_pixel),[bi_message_l
ength  de2bi(mode,2)]);
   half=0; %bitwise XOR of arguments
   [mod_P(:,:,mode) ec1]=data_embedding(re_P,half,mode,message(1:round(end/2)));
   half=1;
   [mod_P(:,:,mode), ec2]=data_embedding(mod_P(:,:,mode),half,mode,message(ec1+1:end));

   if ec1+ec2 == total_message_length
      disp(['PSNR is ' num2str(psnr(original_image,mod_P(:,:,mode),255))])
      result(mode)=psnr(original_image,mod_P(:,:,mode),255);
   else
      disp('failed')
   end
end


%Find predictor which gives maxmimum psnr
[max_result,max_index]=max(result);
if max_result ~= 0 %not equal to
   embedding_Image=mod_P(:,:,max_index);
else
   embedding_Image=[];
   disp('failed')
end
```

5.1.3 preprocess_image.m

This function covers the preprocessing of image to overcome the overflow and underflow conditions. While preprocessing, the changes are recorded using location map.

```
function [m_image loc_map location_map_length]=preproces_image(image)
m_image=image;
[n m]=size(m_image);
counter=0;
location_map_temp=zeros(1,n*m);
for i1=2:n-1  % representing rows
   for j1=2:m-1 % representing columns
      if m_image(i1,j1)==0 || m_image(i1,j1)==255
         counter=counter+1;
         if m_image(i1,j1)==0
            m_image(i1,j1)=1;
         else
            m_image(i1,j1)=254;
         end
         location_map_temp(counter)=1;
      elseif m_image(i1,j1)==254 || m_image(i1,j1)==1
         counter=counter+1;
         location_map_temp(counter)=0;
      end
   end
end
% counter
location_map=location_map_temp(1:counter);
length_loc_map_max=ceil(log2((n-2)*(m-2))); %rounds the value to the nearest integer
if  isempty(location_map)
   loc_map=[];
   location_map_length=de2bi(0,length_loc_map_max); %converts to non-negative integer to binary row
vector
   length_vector1=char(zeros(1,length_loc_map_max)+48);
else
   xE=cell(1,1); %creates 1-by-1 cell array of empty matrices
   xE{1}=[location_map];

   y4=arith07(xE);

   loc_map=reshape(de2bi(y4,8),1,[]); %reshapes the y4 value to 1 column
   location_map_length=de2bi(length(loc_map),length_loc_map_max);

end
```

5.1.4 data_embedding.m

This function covers the embedding process. The data embedding will be carried out using three difference predictors. Each predictor differs in the way they are choosing the higher and lower predictor values. Final embedding output will be decided based on MSE and PSNR values obtained from all the three predictors.

```
function [mod_P,ec]=data_embedding(P,half,pred_method,message,varargin)
if nargin == 4
```

```
    [n m]=size(P);
    LC=zeros(n,m);
    counter=0;
    for i1=2:n-1
        for j1=2:m-1
            if rem(i1+j1,2)==half
                counter=counter+1;
                W=P(i1,j1-1);N=P(i1-1,j1);E=P(i1,j1+1);S=P(i1+1,j1);
                LC(i1,j1)=abs(N-E)+abs(E-S)+abs(S-W)+abs(W-N)+abs(N-S)+abs(E-W);
            end
        end
    end
    pixel_profile=zeros(counter,7);
    counter=0;

    for i1=2:n-1
        for j1=2:m-1
            if rem(i1+j1,2)==half
                counter=counter+1;
                W=P(i1,j1-1);N=P(i1-1,j1);E=P(i1,j1+1);S=P(i1+1,j1);
                W_LC=LC(i1,j1)+LC(i1-1,j1-1)+LC(i1-1,j1+1)+LC(i1+1,j1-1)+LC(i1+1,j1+1);
                pixel_profile(counter,:)=[W_LC P(i1,j1) N W S E (j1-1)*n+i1];
            end
        end
    end
    sorted_pixel_profile=sortrows(pixel_profile,1);
else
    sorted_pixel_profile=varargin{1};
end
message_length=length(message);
counter=length(sorted_pixel_profile);
location=zeros(1,counter);
mod_P=P; %mod_P=image
ec=0;
final_set=0;
temp=zeros(1,counter);

for i1=1:counter
    current_set=sorted_pixel_profile(i1,:);
    loc_y=current_set(7);
    if loc_y > floor(log2(n*m))+2
        %Prediction
        if pred_method==1
            sorted_pixels=sort(current_set(3:6),'descend');
            Pred_h=round(sum(sorted_pixels(1)));
            Pred_l=round(sum(sorted_pixels(4)));
        elseif pred_method==2
            sorted_pixels=sort(current_set(3:6),'descend');
            Pred_h=round(sum(sorted_pixels(1:2))/2);
            Pred_l=round(sum(sorted_pixels(3:4))/2);
        elseif pred_method==3
            sorted_pixels=sort(current_set(3:6),'descend');
            Pred_h=round(sum(sorted_pixels(1:3))/3);
            Pred_l=round(sum(sorted_pixels(2:4))/3);
        elseif pred_method==4
            Pred_h=round(sum(current_set(3:6))/4);
```

```
            Pred_l=Pred_h-1;
        end
        if Pred_h == Pred_l
            Pred_l=Pred_l-1;
        end
        %

        pe=mod_P(loc_y)-Pred_h;
        pixel=mod_P(loc_y);
        if message_length >= ec+1
            [ec, pixel]=prediction(ec,pixel,message(ec+1),pe,1);
            pe=pixel-Pred_l;

            if message_length ~= ec
                [ec, pixel]=prediction(ec,pixel,message(ec+1),pe,-1);
            else
                [ec, pixel]=prediction(ec,pixel,1,pe,-1);
                if message_length ~= ec
                    ec=ec-1;
                end
            end
        else
            break
        end
        location(i1)=loc_y;
        temp(i1)=pixel;
        final_set=i1;
    end
end
if final_set~=0
    mod_P(location(1:final_set))=temp(1:final_set);
end
end

function [ec, pixel]=prediction(ec,pixel,m,pe,direction)
if direction ==1
    if pe == 0
        ec=ec+1;
        pixel=pixel+m;
    elseif pe > 0
        pixel=pixel+1;
    end
else
    if pe == 0
        ec=ec+1;
        pixel=pixel-m;
    elseif pe < 0
        pixel=pixel-1;
    end
end

end
```

5.1.5 grs2rgb.m

This function will convert grey images to rgb images.

```
function res = grs2rgb(img, map)
if nargin<1
   error('grs2rgb:missingImage','Specify the name or the matrix of the image');
end;
if ~exist('map','var') || isempty(map)
   map = hot(64);
end;
[l,w] = size(map);
if w~=3
   error('grs2rgb:wrongColormap','Colormap matrix must contain 3 columns');
end;
if ischar(img)
   a = imread(img);
elseif isnumeric(img)
   a = img;
else
   error('grs2rgb:wrongImageFormat','Image format: must be name or matrix');
end;
% Calculate the indices of the colormap matrix
a = double(a);
a(a==0) = 1; % Needed to produce nonzero index of the colormap matrix
ci = ceil(l*a/max(a(:)));
% Colors in the new image
[il,iw] = size(a);
r = zeros(il,iw);
g = zeros(il,iw);
b = zeros(il,iw);
r(:) = map(ci,1);
g(:) = map(ci,2);
b(:) = map(ci,3);
% New image
res = zeros(il,iw,3);
res(:,:,1) = r;
res(:,:,2) = g;
res(:,:,3) = b;
```

5.1.5 main_decode.m

The main decode function involves re-preprocessing of the stego image. It will undo the side information stored as a location map and extracting the secret text that is embedded in the stego image.

```
function [re_original, re_message]=main_decode(embedding_Image)

[re_mod_P2, re_message2]=data_extraction(embedding_Image,1);
[re_mod_P1, re_message1]=data_extraction(re_mod_P2,0);
re_message=[re_message1 re_message2];

%Undo preprocessing
```

```
[re_n re_m]=size(embedding_Image);
re_message_length_max=floor(log2(re_n*re_m));
re_length_loc_map_max=ceil(log2((re_n-2)*(re_m-2)));

%side information
re_border_pixel=re_message(1:re_message_length_max+2);

start_pos=re_message_length_max+2+1;
re_location_map_length=bi2de(re_message(start_pos:start_pos+re_length_loc_map_max-1));

start_pos=start_pos+re_length_loc_map_max;
re_location_map=re_message(start_pos:start_pos+re_location_map_length-1);
re_message(1:start_pos+re_location_map_length-1)=[];
%Recover border pixel
re_mod_P1(1:re_message_length_max+2)=bitxor(bitxor(re_mod_P1(1:re_message_length_max+2),mod(
re_mod_P1(1:re_message_length_max+2),2)),re_border_pixel);
if isempty(re_location_map)
   re_original=re_mod_P1;
else
   [re_original]=re_preproces_image(re_mod_P1,re_location_map);
end
```

5.1.6 re-preprocess_image.m

In this function, re-preprocessing is taking place based on the location map. The entries in the location map denotes the pixel positions that is changed while preprocessing the image.

```
function [preprocess_image]=re_preproces_image(preprocess_image,location_map)
[n m]=size(preprocess_image);

xC = arith07(bi2de(reshape(location_map,[],8)));
re_location_map=xC{1};
counter=1;

for i1=2:n-1
   for j1=2:m-1
      if counter < length(re_location_map)
         if preprocess_image(i1,j1)==1 || preprocess_image(i1,j1)==254
            if re_location_map(counter)==1
               if preprocess_image(i1,j1)==1
                  preprocess_image(i1,j1)=0;
               else
                  preprocess_image(i1,j1)=255;
               end
               counter=counter+1;
            end
         end
      end
   end
end
% counter
```

5.1.7 data_extraction.m

Data extraction is carried out in this function using three different predictors. The higher and lower predictor values will be different for each predictor. Final predictor selection is based on the higher PSNR values and lower MSE values.

```
function [mod_P, re_message]=data_extraction(P,half)
[n m]=size(P);
%border side information extraction
message_length_max=floor(log2(n*m));
b_side_info = P(1:message_length_max+2);
bi_message_length=bi2de(mod(b_side_info(1:end-2),2));
pred_method=bi2de(mod(b_side_info(end-1:end),2));


LC=zeros(n,m);
counter=0;
for i1=2:n-1
   for j1=2:m-1
      if rem(i1+j1,2)==half
         counter=counter+1;
         W=P(i1,j1-1);N=P(i1-1,j1);E=P(i1,j1+1);S=P(i1+1,j1);
         LC(i1,j1)=abs(N-E)+abs(E-S)+abs(S-W)+abs(W-N)+abs(N-S)+abs(E-W);
      end
   end
end
pixel_profile=zeros(counter,7);
counter=0;

for i1=2:n-1
   for j1=2:m-1
      if rem(i1+j1,2)==half
         counter=counter+1;
         W=P(i1,j1-1);N=P(i1-1,j1);E=P(i1,j1+1);S=P(i1+1,j1);
         W_LC=LC(i1,j1)+LC(i1-1,j1-1)+LC(i1-1,j1+1)+LC(i1+1,j1-1)+LC(i1+1,j1+1);
         pixel_profile(counter,:)=[W_LC P(i1,j1) N W S E (j1-1)*n+i1];
      end
   end
end
sorted_pixel_profile=sortrows(pixel_profile,1);

if half == 1
   message_length=bi_message_length-round(bi_message_length/2);
else
   message_length=round(bi_message_length/2);
end

counter=length(sorted_pixel_profile);
location=zeros(1,counter);
mod_P=P;
ec=0;
final_set=0;
temp=zeros(1,counter);
```

```
for i1=1:counter
    current_set=sorted_pixel_profile(i1,:);
    loc_y=current_set(7);
    if loc_y > message_length_max+2
        %Prediction
        if pred_method==1
            sorted_pixels=sort(current_set(3:6),'descend');
            Pred_h=round(sum(sorted_pixels(1)));
            Pred_l=round(sum(sorted_pixels(4)));
        elseif pred_method==2
            sorted_pixels=sort(current_set(3:6),'descend');
            Pred_h=round(sum(sorted_pixels(1:2))/2);
            Pred_l=round(sum(sorted_pixels(3:4))/2);
        elseif pred_method==3
            sorted_pixels=sort(current_set(3:6),'descend');
            Pred_h=round(sum(sorted_pixels(1:3))/3);
            Pred_l=round(sum(sorted_pixels(2:4))/3);
        elseif pred_method==4
            Pred_h=round(sum(current_set(3:6))/4);
            Pred_l=Pred_h-1;
        else
            pause
        end
        if Pred_h==Pred_l
            Pred_l=Pred_l-1;
        end

        pe=mod_P(loc_y)-Pred_l;
        pixel=mod_P(loc_y);
        if message_length >= ec+1
            [ec, pixel, re_m]=prediction(ec,pixel,pe,-1);
            pe=pixel-Pred_h;
            if isempty(re_m)==0
                re_message(ec)=re_m;
            end
            if message_length ~= ec
                [ec, pixel, re_m]=prediction(ec,pixel,pe,1);
                if isempty(re_m)==0
                    re_message(ec)=re_m;
                end
            else
                [ec, pixel, re_m]=prediction(ec,pixel,pe,1);
                if message_length ~= ec
                    ec=ec-1;
                end
            end
        else
            break
        end
        location(i1)=loc_y;
        temp(i1)=pixel;
        final_set=i1;
    end
end
if final_set~=0
    mod_P(location(1:final_set))=temp(1:final_set);
```

```
end

end

function [ec, pixel,re_m]=prediction(ec,pixel,pe,direction)
re_m=[];
if direction ==1
   if pe == 0
      ec=ec+1;
      re_m=0;
   elseif pe == 1
      ec=ec+1;
      re_m=1;
      pixel=pixel-1;
   elseif pe > 1
      pixel=pixel-1;
   end
else
   if pe == 0
      ec=ec+1;
      re_m=0;
   elseif pe == -1
      ec=ec+1;
      re_m=1;
      pixel=pixel+1;
   elseif pe < -1
      pixel=pixel+1;
   end
end
end
```

5.1.8 structuralsimilarity.m

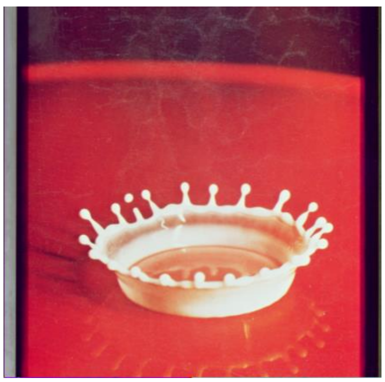
This function calculates the structural similarity index and it returns the information in an image that is the same size as the image whose quality is being measured.

```
function structure_similarity=structuralsimilarity(images,final)
k1=0.01;
k2=0.03;
l=255;
c1=square(k1.*l);
c2=square(k2.*l);
[A,B,C]=size(images);
%for i=1:A
b1=images(1,:,:);
I1=b1(1,1);
I=I1;
X=double(I);
[o,n]=size(X)
%for j=1:A
b2=final(1,:,:);
I3=b2(1,1);
I2=I3{1,:,:};
```

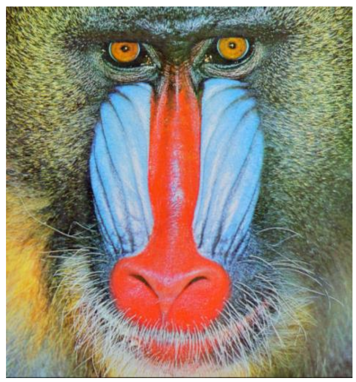
```
Y=double(I2);
[p,q]=size(Y)
xme=mean(X(:));
yme=mean(Y(:));
xv=var(X(:));
yv=var(Y(:));
cxy1=cov(X(:),Y(:));
cxy=cxy1(1,2);
structure_similarity(1,1)=(((2.*xme.*yme)+c1).*((2.*cxy)+c2)./((square(xme)+square(yme)+c1).*(xv+yv+c2
)));

%end
%end
end
```

5.2 SAMPLE OUTPUT



4.2.01.tiff (Splash)



4.2.03.tiff (Baboon)

4.2.04.tiff (Lena)

4.2.05.tiff (F-16)

4.2.06.tiff (Boat)

Fig 5.2.1 TEST IMAGES

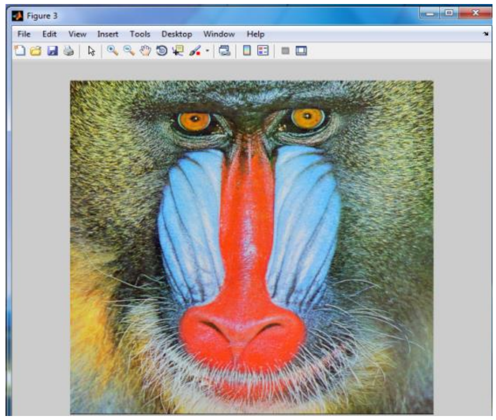


Fig 5.2.2 Cover Image

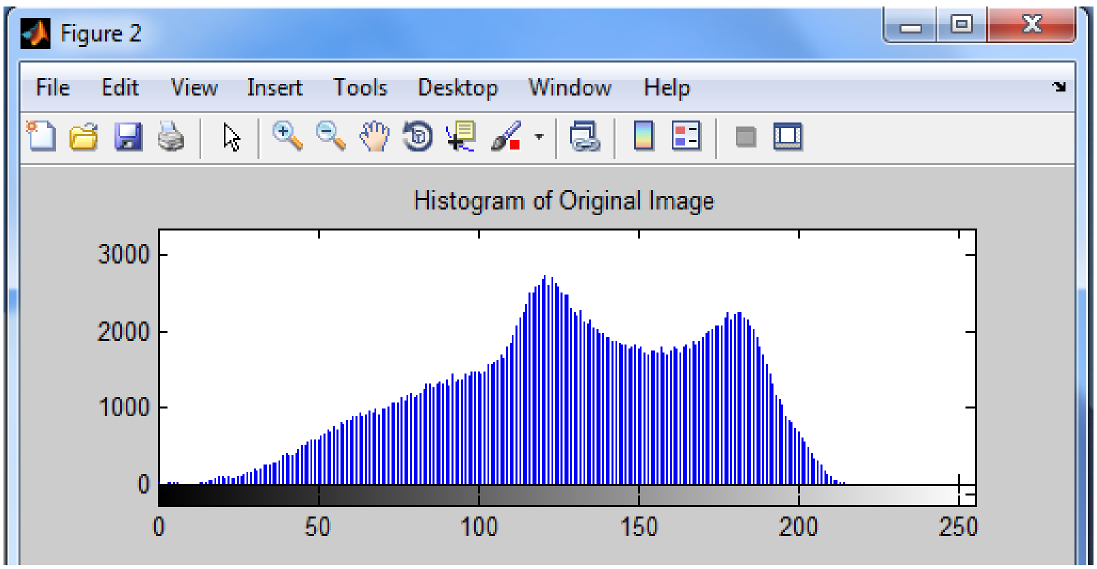


Fig 5.2.3 Histogram Of Original Image

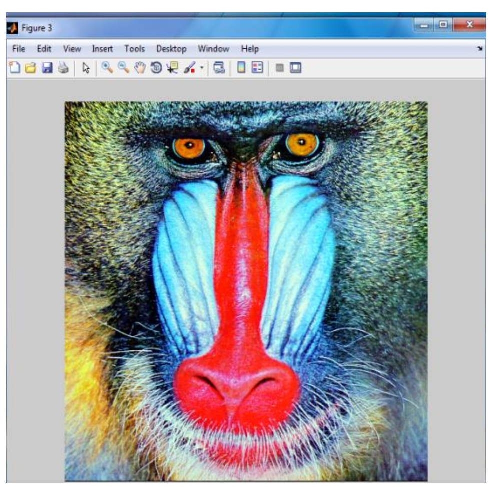


Fig 5.2.4 Embedded Image

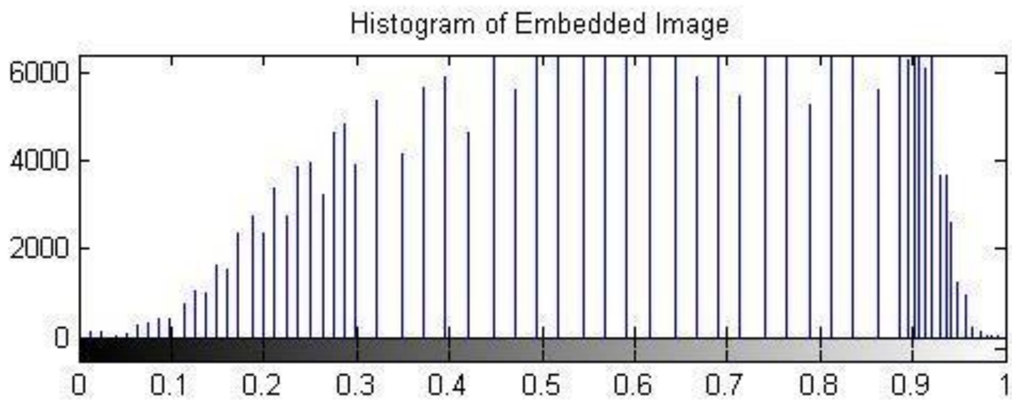


Fig 5.2.5 Histogram of Embedded Image

5.3 PARAMETERS SETTING:

The Mean Square Error (**MSE)** represents the cumulative squared error between the compressed and the original image. Peak Signal-to-Noise Ratio (**PSNR**) denotes the ratio between the maximum possible power of a signal and the power of corrupting noise.

The lower the value of **MSE**, the lower is the error and higher is the **PSNR** value. For 16-bit data typical values for the **PSNR** are between 60 and 80 dB.

The Structural Similarity Index (**SSIM**) is a perceptual metric that quantifies image quality degradation caused by processing such as data compression or by losses in data transmission.

5.4 EXPERIMENTAL RESULTS

5.4.1 EXISTING METHOD:

For <u>4.2.03.tiff (Baboon)</u>

Embedding Capacity- 20,000 bits

TABLE 5.4.1: Statistical evaluation of MSE, PSNR and SSIM in existing method.

METHOD	MSE	PSNR	SSIM
LSB Embedding	0.00075	80.607	0.99979
[5]	0.00958	77.859	0.92098
[6]	0.00991	79.865	0.94367

5.4.2 PROPOSED METHOD:

Statistical evaluation (mean) on USC-SIPI images

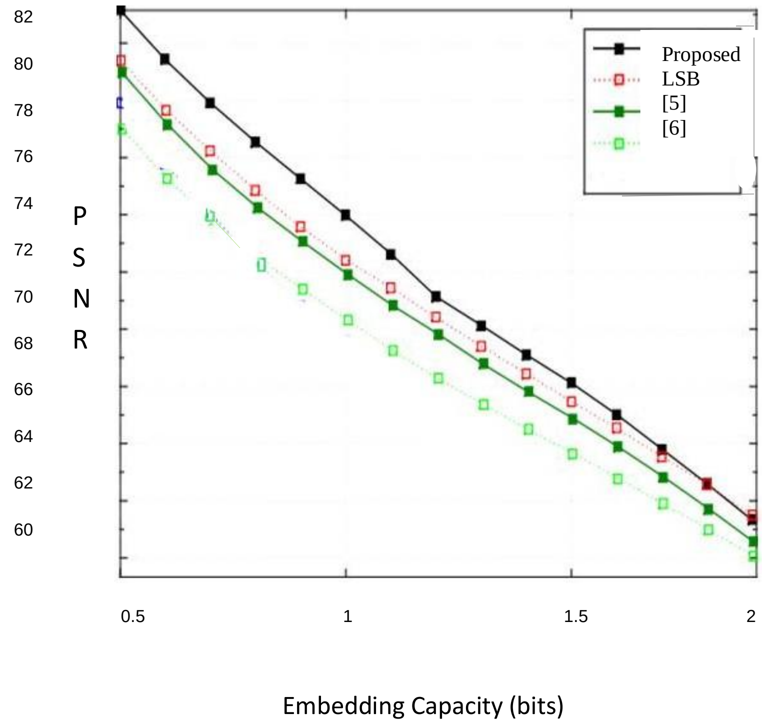
Embedding Capacity- 20,000 bits

**Image Dimension - ** 512*512

TABLE 5.4.2 : Statistical evaluation of MSE, PSNR and SSIM in proposed method.

IMAGE	MSE	PSNR	SSIM
4.2.01.tiff (Splash)	0.00010	81.0026	0.99932
4.2.03.tiff (Baboon)	0.00056	82.6136	0.99983
4.2.04.tiff (Lena)	0.00095	79.437	0.99973
4.2.05.tiff (F-16)	0.00095	79.569	0.99984
4.2.06.tiff (Boat)	0.00022	80.5347	0.99925

5.4.3 PERFORMANCE COMPARISON:



CHAPTER 6

CONCLUSION

In this project, we have proposed a new reversible data hiding method with image contrast enhancement. A preprocessing strategy has been developed by merging the adjacent rows or columns of bins in two-dimensional histogram while the changes that have been made can be recorded. Since the constraints on choosing the bins have been relaxed in two-dimensional plane, the lowest bins are chosen in preprocessing while the highest bins can be expanded for data embedding. Thus comparable contrast enhancement effects can be obtained with satisfactory image quality. Different from the normal methods, the process of image contrast enhancement can be performed in a lossless manner with the proposed method. The experimental results have shown that an original image can be exactly recovered from a series of its contrast-enhanced versions. Moreover, the degree of contrast enhancement can be adjusted with respect to only one parameter. Compared with the methods using one-dimensional histogram, generally better performances in contrast enhancement effects and image quality have been achieved based on two-dimensional histogram modification.

CHAPTER 7

REFERENCES

- **[1]** HAO-TIAN WU, WEIQI MAI, 2019, **Reversible Data Hiding with image contrast enhancement based on Two-Dimensional histogram Modification.** IEEE Access.

- **[2]** ZHENXING QIAN, XINPEN ZHANG**,** 2019**, New Framework of Reversible Data Hiding in Encrypted JPEG Bitstreams.** IEEE Access (Volume: 7).

- **[3]** V.R.VIJAY KUMAR, V.SURESH, 2017, **Histogram shifting based Reversible Data Hiding with controlled contrast enhancement.** IEEE Third International Conference on Biosignals, Images and Instrumentation (ICBSII).

- **[4]** HUSSAIN, NYEEM, 2017, **Reversible Data Hiding with image bit-plane slicing.** IEEE 20th International Conference of Computer and Information Technology Instrument.

- **[5]** QICHAO YING, ZHENXING QIAN, 2019, **Reversible Data Hiding with image enhancement using histogram shifting.** IEEE Access (Volume: 7).

- **[6]** HAOTIAN WU, JEANLUC DUGELAY, 2015, **Reversible Image Data Hiding with contrast enhancement.** IEEE Signal Processing Letters (Volume:22).

- **[7]** SUAH KIM, ROLF LUSSI, XIAOCHAO QU, 2015 **Automatic contrast enhancement using Reversible Data Hiding.** IEEE International Workshop on Information Forensics and Security (WIFS).

- **[8]** YU-CHI CHEN, TSUNG HSUAN, HUNG, 2019, **A new Reversible Data Hiding in encrypted image based on multi-secret sharing and lightweight cryptographic algorithm.** IEEE Transactions on Information Forensics and Security (Volume: 14 , Issue: 12).

- **[9]** WEI ZHANG, PING KONG, 2018, **Real time Reversible Data Hiding in encrypted images based on hybrid embedding mechanism.** Journal of Real-Time Image Processing.
- **[10]** SHAOHUA TANG, ZHIYUAN YANG, 2019, **High capacity Reversible Data Hiding in encrypted images by bit plane partition and MSB prediction.** IEEE Access (Volume: 7).
- **[11]** ZHAOXIA YIN, XINPENG ZHANG, 2020, **Reversible Data Hiding in encrypted images based on multi-MSB prediction and Huffman Coding.** IEEE Transactions on Multimedia (Volume:22).
- **[12]** SIREN CAI, XINLU GUI, 2013, **An efficient Reversible Data Hiding scheme based on Reference pixel and Block selection.** IEEE 2013 Ninth International Conference on Intelligent Information Hiding and Multimedia Signal Processing.
- **[13]** SUAH KIM, ROLF LUSSI, 2019, **Reversible Data Hiding with Automatic Brightness preserving contrast enhancement.** IEEE Transactions on Circuits and Systems for Video Technology (Volume:29).
- **[14]** HAOTIAN WU, JEANLUC DUGELAY, 2015, **Reversible Data Hiding using controlled contrast enhacement and Integer Wavelet Transform.** IEEE Signal Processing Letters (Volume: 22 , Issue: 11).
- **[15]** YINGHUI XIA, HAOTIAN WU, 2016, **Reversible Data Hiding in Medical Image for Contrast Enhancement of ROI.** IEEE 12th International Conference on Computational Intelligence and Security (CIS).
- **[16]** VINIT AGHAM, TAREEK PATTEWAR, 2014, **A novel approach towards separable Reversible Data Hiding technique.** IEEE International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT).