DESIGN OF DNA BASED COLOR IMAGES CRYPTOSYSTEM AND ITS SECURITY ANALYSIS

BY

MOHD ZAIN

(Admission No. 22MT0214)

THESIS

SUBMITTED TO

INDIAN INSTITUTE OF TECHNOLOGY

(INDIAN SCHOOL OF MINES), DHANBAD

For the award of the degree of

MASTER OF TECHNOLOGY

MAY, 2024

INDINDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES) DHANBAD

CERTIFICATE FOR THE FINAL VERSION OF DISSERTATION

This is to certify that the Dissertation entitled **"Design of DNA Based Color Images Cryptosystem and its Security Analysis"** being submitted to the Indian Institute of Technology (Indian School of Mines), **Mr. Mohd Zain**, Admission No. **22MT0214** for the award of the Degree of **Master of Technology** from IIT (ISM), Dhanbad, is a bonafide work carried out by him, in the Department of **Mathematics and Computing**, IIT (ISM), Dhanbad, under my supervision and guidance. The dissertation has fulfilled all the requirements as per the regulations of this Institute and, in my opinion, has reached the standard needed for submission. The results embodied in this dissertation have not been submitted to any other university or institute for the award of any degree or diploma.

Signature of Supervisor (s)

Name: **Prof. Abhay Kumar Singh**

Date: 08 May, 2024

INDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES) DHANBAD

DECLARATION BY THE STUDENT

I hereby declare that the work which is being presented in this dissertation entitled "**Design of DNA Based Color Images Cryptosystem and its Security Analysis**" in partial fulfilment of the requirements for the award of the degree of **Master of Technology** in **Data Analytics** is an authentic record of my own work carried out during the period from **01/08/2023** to **10/05/2024** under the supervision of **Prof. Abhay Kumar Singh** Department of **Mathematics and Computing**, Indian Institute of Technology (ISM) Dhanbad, Jharkhand, India.

I acknowledge that I have read and understood the UGC (Promotion of Academic Integrity and Prevention of Plagiarism in Higher Educational Institutions) Regulations, 2018. These Regulations were published in the Indian Official Gazette on 31st July, 2018.

I confirm that this Dissertation has been checked for plagiarism using the online plagiarism checking software provided by the Institute. At the end of the Dissertation, a copy of the summary report demonstrating similarities in content and its potential source (if any) generated online using plagiarism checking software is enclosed. I herewith confirm that the Dissertation has less than 10% similarity according to the plagiarism checking software's report and meets the MoE/UGC Regulations as well as the Institute's rules for plagiarism.

I further declare that no portion of the dissertation or its data will be published without the Institute's or Guide's permission. I have not previously applied for any other degree or award using the topics and findings described in my dissertation.

(Signature of the Student)

Name of the Student: Mohd Zain

Admission No.: 22MT0214

Department: Mathematics and Computing

INDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES) DHANBAD

CERTIFICATE FOR CLASSIFIED DATA

(To be submitted at the time of Final Dissertation Submission)

This is to certify that the Dissertation entitled "**Design of DNA Based Color Images Cryptosystem and its Security Analysis**" being submitted to the Indian Institute of Technology (Indian School of Mines), Dhanbad by **Mr Mohd Zain** for award of Master Degree in **Data Analytics** does not contain any classified information. This work is original and yet not been submitted to any institution or university for the award of any degree.

Signature of Supervisor(s)

Signature of Student

INDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES) DHANBAD

CERTIFICATE FOR ENGLISH CHECKING

This is to certify that the Dissertation entitled "**Design of DNA Based Color Images Cryptosystem and its Security Analysis**" being submitted to the Indian Institute of Technology (Indian School of Mines), Dhanbad, by **Mr Mohd Zain** , Admission No **22MT0214** for the award of the **Degree of Master of Technology** has been thoroughly checked for quality of English and logical sequencing of topics.

It is hereby certified that the standard of English is good and that grammar and typos have been thoroughly checked.

Signature of the Guide (s)

Name: Prof. Abhay Kumar Singh

Date: 08 May, 2024

Signature of the Student

Name: Mohd Zain

Date: 08 May, 2024

INDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES) DHANBAD

COPYRIGHT AND CONSENT FORM

To ensure uniformity of treatment among all contributors, other forms may not be substituted for this form, nor may any wording of the form be changed. This form is intended for original material submitted to the IIT (ISM), Dhanbadand must accompany any such material to be published by the ISM. Please read the form carefully and keepa copy for your files.

TITLE OF DISSERTATION: Design of DNA Based Color Images Cryptosystem and its Security Analysis

AUTHOR'S NAME & ADDRESS: Mohd Zain & Bijnor, Uttar Pradesh, INDIA

COPYRIGHT TRANSFER

1. The undersigned hereby assigns to the Indian Institute of Technology (Indian School of Mines), Dhanbad all rights under copyright that may exist in and to: (a) the above Work, including any revised or expanded derivative works submitted to the ISM by the undersigned based on the work; and (b) any associated written or multimedia components or other enhancements accompanying the work.

CONSENT AND RELEASE

2. In the event the undersigned makes a presentation based upon the work at a conference hosted or sponsored in whole or in part by the IIT (ISM) Dhanbad, the undersigned, in consideration for his/her participation in the conference, hereby grants the ISM the unlimited, worldwide, irrevocable permission to use, distribute, publish, license, exhibit, record, digitize, broadcast, reproduce and archive; in any format or medium, whether now known or hereafter developed: (a) his/her presentation and comments at the conference; (b) any written materials or multimedia files used in connection with his/her presentation; and (c) any recorded interviews of him/her (collectively, the "Presentation"). The permission granted includes the transcription and reproduction of the Presentation for inclusion in products sold or distributed by IIT(ISM) Dhanbad and live or recorded broadcast of the Presentation during or after the conference.
3. In connection with the permission granted in Section 2, the undersigned hereby grants IIT (ISM) Dhanbad the unlimited, worldwide, irrevocable right to use his/her name, picture, likeness, voice and biographical information as part of the advertisement, distribution and sale of products incorporating the Work or Presentation, and releasesIIT (ISM) Dhanbad from any claim based on right of privacy or publicity.
4. The undersigned hereby warrants that the Work and Presentation (collectively, the "Materials") are original and that he/she is the author of the Materials. To the extent the Materials incorporate text passages, figures, data or other material from the works of others, the undersigned has obtained any necessary permissions. Where necessary, the undersigned has obtained all third-party permissions and consents to grant the license above and has provided copies of such permissions and consents to IIT (ISM) Dhanbad.

GENERAL TERMS

* The undersigned represents that he/she has the power and authority to make and execute this assignment.
* The undersigned agrees to indemnify and hold harmless the IIT (ISM) Dhanbad from any damage or expense thatmay arise in the event of a breach of any of the warranties set forth above.
* In the event the above work is not accepted and published by the IIT (ISM) Dhanbad or is withdrawn by the author(s) before acceptance by the IIT(ISM) Dhanbad, the foregoing copyright transfer shall become null and void and all materials embodying the Work submitted to the IIT(ISM) Dhanbad will be destroyed.
* For jointly authored Works, all joint authors should sign, or one of the authors should sign as authorized agent forthe others.

Signature of the Student

ACKNOWLEDGEMENTS

I would like to convey my gratitude to my supervisor, **Prof. Abhay Kumar Singh**, for his advice and assistance in supervising the research work mentioned in this thesis. I am thankful for his perseverance, insights, passion, and suggestions, as well as for consistently encouraging me to give my best. I would like to express my gratefulness to **Prof Ranjit Kumar Upadhyay**, Head of Department, Department of Mathematics and Computing, IIT (ISM) Dhanbad for their support. I would like to thank all the professors and staff members of the Department of Mathematics and Computing, IIT (ISM) Dhanbad, for their direct and indirect assistance in completing my study. In addition, I am thankful to the Government of India for giving me financial support in the form of a scholarship for my Postgraduate work. Finally, I want to express my gratitude to my parents and friends for their constant support and encouragement.

Date: 08 May, 2024

Place: IIT (ISM), Dhanbad (Mohd Zain)

Table of Contents

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The goal of image cryptosystems is to protect image transmission when there are network adversaries present. To ensure secrecy, images are subject to encryption to produce unintelligible cipher images; the techniques used for this process differ significantly from those applied to text data. The majority of the cryptosystem consider complicated or confusion–diffusion architectures that changes and permute the values of the pixels. Their aim is to make the relation between key and text as complex as possibleand to propagate every text knowledge to the entire system. These frequently entail binary operations like bitwise XOR, plus–minus, DNA operations, etc; are carried out utilising chaotic maps, each of which have certain limitations. This thesis employs a binary function that can be applied to both traditional and DNA techniques for coloured natural images, and can be applied to any kind of image cryptosystem. Each of the color component of the image follows some steps which start with inter channel mixing and mix rows which takes the value from initial vector and mixes it with the first element of every row and propagates its value to the entire row operating wth the suggested operator , then Arnold cat map algorithm is applied to shuffle the pixels. Secondly spiral mixing of the pixel is applied in four different directions viz. forward row and column, backward row and column to prevent differential attack. Finally, encoding, substitution, and decoding based on DNA is carried out. A multiple collapse chaotic map is used to derive initialization vector, rule charts and DNA substitution map which are used in encoding and decoding processes. Analysis and experimental findings shows that our cryptosystem has significant performance and different metrics shows it can withstand different type of attacks. Comparative analysis with modern cryptosystems demonstrate the acceptable performance of our cryptosystem.

Chapter 1: INTRODUCTION

1.1 Introduction

The ease with which technology is now available, such as smartphones and the Internet, has made sharing images quite simple. These days, a growing number of multimedia images are routinely sent across public networks, and there is an increasing focus on how to safeguard them. Between sending and receiving the images there is along path where the vulnerabilities are present. The wrong doers find these weaknessess in the system and exploit them to compromise the network. Several measures have been taken to decrease these exploitations in the network. One of the best techniques is picture encryption, which converts the important images into noise-like ones that can be sufficient to protect the hidden images. The plain images can easily be understood by the humans by merely seeing them but the crypted image is distorted image where the pixel are look like the dot which cannot be understood by any humam as well as any computer system. The crypted images are less prone to attack as it cannot represent any content inside it. In contrast to text data, images are stored in several formats based on their intended use and have highly associated properties. Both lossless and lossy formats are used to store natural images. The systems which helps in making image crypted are known as cryptosystems. Cryptosystems intended for images are distinct from those used for other purposes since they are made to change pixel intensity values and guarantee the removal of statistical and visual information.

The area of study that deals with information, images security is cryptography. It includes methods for guaranteeing data accuracy and privacy, especially when data is sent or kept in a manner that may make it susceptible to interception. It is the process of transforming data from one form—such as readable to unreadable. The primary objective of cryptography is to transmit information over an unprotected channel between a sender and a recipient in a way that prevents a malevolent user from reading the actual material. In its modern form, cryptography is essential for digital security, enabling everything from secure online transactions to private communications over the internet.

The key concepts of cryptography are:

- **Plaintext or Plain Image:** The original and understandable message from the user to the recipient.
- **Encryption:** The layer of protection given to text as well as images. Transforming plaintext into a scrambled format (ciphertext) to make it unreadable using some security key that can only be understood if decrypted.
- **Decryption:** The word done at the recipient side for turning ciphertext or cipher image back into readable plaintext or initial image using a secret key.
- **Key:** A hexadecimal or binary values generally of a long length used in the encryption and decryption processes. The idea or key to give a strength to encryption as well as to make the decryption easier.
- **Hashing:** This technique is used to store the information crypted some where such as databases, password stored in social sites.

The cryptography is processed in generally two steps viz. encryption and decrytion. Generally at user side when sender sends a message then that send button only contains the process of encryption. The crypted text or image then travelled in the path. At recievers side before user open it, it is decrypted using the decryption process.

Below is the flow graph of this entire process:

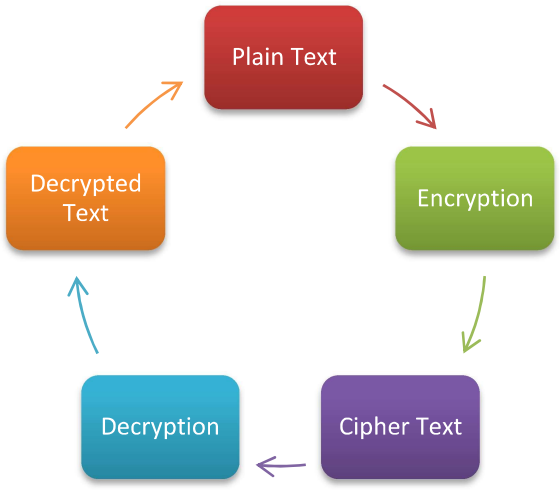


Fig 1.1 Basic Idea of process of cryptosystems

These processes require other events also such as key, different mappings etc.

The Techniques used in the cryptography are merely of two types:

- **Cryptography using a Symmetric Key**: This method involves a single key that both the sender and receiver use during the processes of encryption and decryption. The shared key simplifies the design of such cryptosystems, making them generally easier and faster to implement compared to their counterparts. In symmetric key cryptography, the primary concern is maintaining the confidentiality of the data. By utilizing the same key, both parties can seamlessly encrypt and decrypt messages, ensuring that the information remains secure from unauthorized access. However, the challenge lies in securely distributing and managing the key so that only authorized entities possess it. The process is shown in fig 1.2
- **Cryptography using Asymmetric Key**: Unlike symmetric key cryptography, asymmetric key cryptography employs two different keys — a public key and a private key. The sender uses the public key for encryption, while the receiver uses their private key for decryption. This method addresses not only the confidentiality of the messages but also their authenticity, ensuring that the communications are both secure and verifiable. The public key can be widely distributed and available to anyone, but the private key must be kept secure by the receiver. This setup is particularly useful for scenarios where secure data transmission is required between parties with no prior shared secrets. Asymmetric cryptography is foundational for digital signatures and key exchange protocols, which are critical for securing modern data communications.

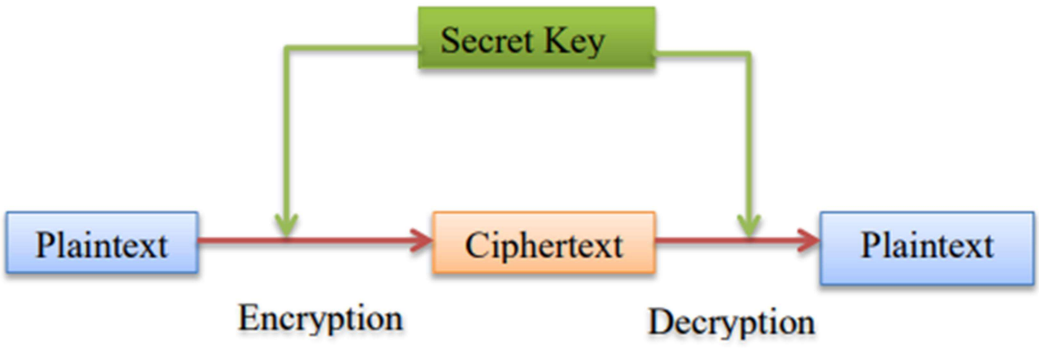


Fig 1.2: Cryptography using Symmetric Key

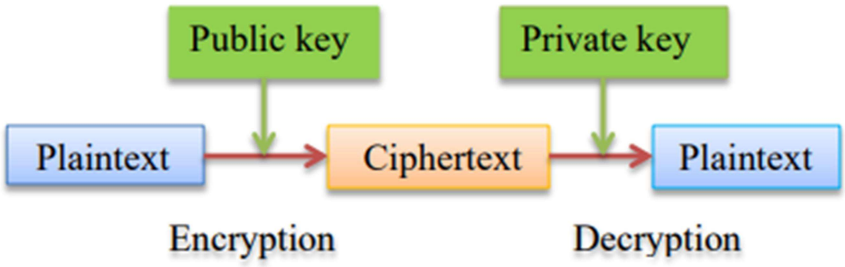


Fig 1.3: Cryptography using Asymmetric Key

1.2 DNA based Cryptosystems

DNA-based color imaging represents an innovative approach in the field of image encryption, where color images are encoded into sequences of DNA. This technique falls under the umbrella of DNA computing, a fascinating area where DNA is utilized not just as the genetic material in living organisms but also as a medium for data storage and computation. Just as traditional computers use binary digits (0s and 1s) to store and process information, DNA computing uses the four nucleotides of DNA—Adenine is (A), Thymine is (T), Cytosine is (C), and Guanine is (G)—to represent data.

In the context of DNA-based color imaging, the colors in an image are not represented by usual pixel values but are instead translated into a sequence of DNA nucleotides. This transformation involves a meticulous process where each pixel value from an image is mapped to a specific sequence of DNA bases. For instance, the red, green, and blue (RGB) values that define the color of each pixel could be represented by distinct combinations of the four DNA bases. This could mean assigning specific nucleotide sequences to various intensities and combinations of the RGB color model. Fig 1.4 illustrate the DNA based cryptosystem's processes.

Overview of Conversion- The value of pixel 0-255 can be converted into DNA sequence. Using a set of predefined rules we can map every two bit of a 8 bit binary sequence to one of the DNA nucleotides. Images are just a 2d array, every element of a matix is one the pixel value. These values are always in range of 0 to 255 which can covered by 8 bit binary representation. Having 8 bit makes it easier to convert this representation into 4 DNA nucleotides. Every two bit of the binary number can be converted to one of the four nucleotides using some mapping function and encoding rules. This process is illustrated in fig 1.5

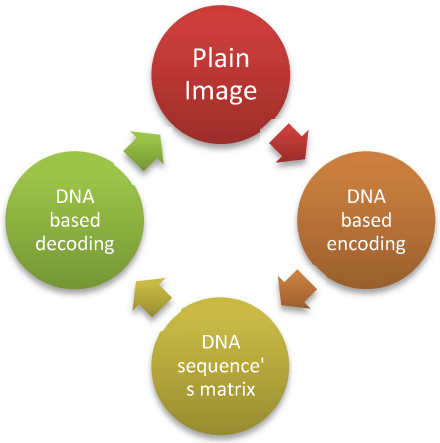


Fig1.4: process of DNA based cryptosystems

25	**54**	**65**
12	78	68
76	98	31

CACA	**CGCG**	**ACAC**
CCTT	ACTG	ACCT
ACTT	ATAG	CATA

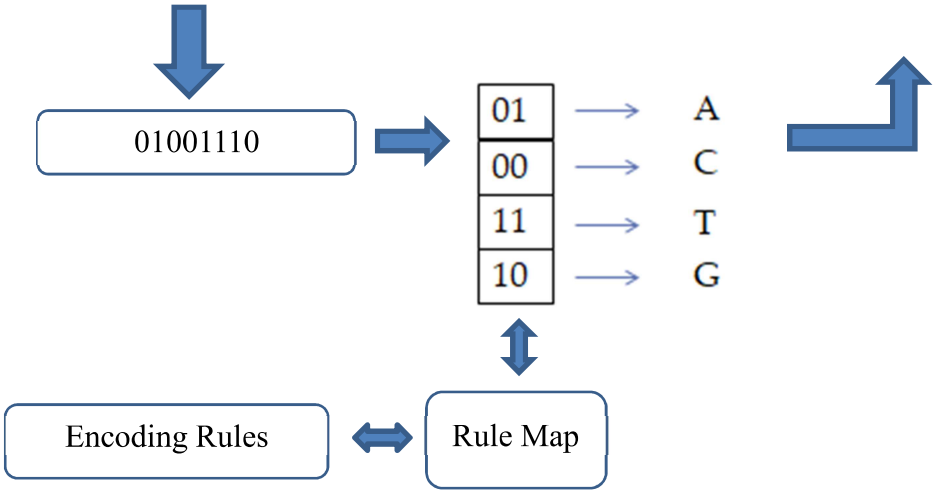


Fig1.5: The process of DNA conversion

The initial part of the process is an image matrix and the latter part is DNA encoded image. There can be different DNA string for same value as it depend on rules choosen by encoding rules principle through rule maps.

1.3 Research Objectives

The goal of this thesis is to expand the suggested operator—which has been effectively applied to single-channel images—to include RGB (Red, Green, and Blue) coloured images. The principal aim is to broaden the efficacy and relevance of the suggested operator for safely encrypting RGB images while maintaining their integrity and confidentiality. In order to verify the Binary operator's performance, experimental finding will be condcuted with various matrics that are required for checking performance, also comparative analysis will be conducted with other operators such as XOR, plus-minus and also with previously defined and implemented coloured image encryption systems.

Key Components:

- **Generalization of Proposed Operator**: The proposed operator has demonstrated promising results in securing single-channel images. We will explore the necessary modifications and enhancements required to make it suitable for RGB colored images, while ensuring that the core security properties of the operator remain intact.
- **Evaluation**: A carefully selected images of RGB colored will be used to evaluate the encryption performance of the proposed generalized operator and the comparison with other systems. The evaluation will be based on objective criteria to ensure a fair and rigorous assessment.
- **Comparative Analysis**: We will conduct a comparative analysis to evaluate the performance of the proposed generalized operator against existing colored image encryption systems. This analysis will involve quantitative metrics to assess security, computational efficiency, and resistance to attacks.

The research aims to contribute to the field of image encryption by extending the applicability of the proposed operator and providing insights into its strengths and limitations compared to other established colored image encryption techniques. This will help to generalize the performance of the operator with the color images as well and to know where the operator stand with color images.

1.4 Problem Statement

A novel techniques of color image cryptosystems with the proposed binary operator. The binary operator that will be used is already defined as:

$$u \otimes v = \{(u+1) * (v+1) \bmod p\} - 1.$$

Where if there is a set of *n* elements then $p = n+1$ for a prime number which means p will always be a prime number.

1.5 Limitations

Creating a DNA-based color image cryptosystem introduces several challenges and limitations. Firstly, the process of encoding color images into DNA sequences is complex and requires specialized techniques, which may limit its practicality and scalability. Additionally, DNA synthesis and sequencing technologies are currently expensive and time-consuming, making large-scale implementation costly and inefficient.

Another limitation is the lack of standardized protocols and algorithms for DNA-based cryptography. Unlike traditional cryptographic methods that have well-established standards and best practices, DNA-based cryptosystems are still in the early stages of development, lacking uniformity and rigor in terms of security and reliability. This lack of standardization may hinder interoperability and adoption across different applications and industries.

1.6 The steps of organizing thesis work

This thesis follows eight chapters which are:

Chapter 1 introduces the details of thesis research area and different apects which are the part of this field. Different processess, types used in topics of this research. This chapter also organizes the content of other chapters as well.

Chapter 2 conducts literature review that was helpful understanding more about the work and research that have already done and currently going in the field of DNA

cryptography. Also literature review highlights the research gap which is present in the developed cryptosystems and in what aspect more work can be done to fill these gaps.

Chapter 3 describes about the proposed cryptosystem and its requirement before proceeding to encryption. It describe about the key, chaotic maps, rule maps, maps for substitution of DNAs and initial array. These maps are required in encoding and decoding while initial array is required for other stages.

Chapter 4 is an Encryption chapter which describe encryption algorithm in details. The encryption algorithm finishes with seven stages in which every stage has some effect on over all procedure.

Chapter 5 contain every detail of decryption procedure and algorithm. Decryption procedure is just the reverse of encryption procedure which requires some changes.

Chapter 6 Experimental Analysis and results have been demonstrated in this chapter. Various metrics are tested for measuring the performance of our cryptosystem. Comparative analysis is also conducted with other operator such as xor and plus-minus to demonstrate the ability of suggested operator with color images in relative to these common operators. Also comparison with standard and in use cryptosystem is also taken into consideration to calculate where our cryptosystem stands.

Chapter 7 concludes the thesis work with summarization and future scope.

Chapter 2: Literature Review

Literature review is a critical component of academic research, providing an overview and evaluation of existing scholarship relevant to a particular topic or research question. It serves several essential purposes in the research process. By examining a range of sources researchers can gain insights into the current state of knowledge and identify areas where further research is needed.

Most image cryptosystems produce quasirandom sequences, wherein the values are predetermined by the initial values supplied to them; these sequences serve as the algorithm's symmetric key, by using specific chaotic maps such as the 2D-Infinite Collapse Map [1], Logistic Maps [2] 3D chen chaotic sysytem [3], Hyper Chaotic Systems [4], Logistic Tent System [5], 4D memristive Hyper Chaos [6], Fractional Order Hyperchaotic System [7], Fractional Fourier Transform [8] etc.

Most of the cryptosystem follows complex architectures,

2.1 Chai Model: Xiuli Chai et al.[3] devised a method of encryption that creates stochastic sequences using the Chen chaotic system, which are then used to form arrays for key stream creation and image permutation. The two stages of their image encryption approach are diffusion and permutation.

2.2 Moattar Model: Mohammad Hossein Moattar and Abolfazl Yaghouti Niyat [4] proposed a cryptosystem in which color image is firstly decomposed into R(Red), G(Green) & B(Blue) channels, permutation is used to mix up all three components. Using DNA encoding rules permuted components are converted into DNA matrices. After this, Diffusion is applied. To enhance security they applied second confusion scheme.

2.3 Liu Model: In paper [6] Zhentao Liu et al. suggested use of 4D memristive hyper chaos with dynamic DNA for colour image encryption in which the process to obtain the R, G, and B DNA matrices, dynamic encoding is applied independently to each of the image's three channels. For all three matrices, dynamic dispersion and dispersal are

utilised. Finally, DNA decoding and component integration resulted in a single encrypted image.

2.4 Farah Model: In the paper[8] Mohd Amine Ben Farah et al. introduced a cryptosystem with the operations like DNA encoding, XOR with encoded image and SHA-2 algorithm, substitution, DNA encoding, Fractional fourier transform applied on encoded image, XOR is applied on transformed image with Chaotic sequence obtained from Lorenz system, again substitution, again Fractional Fourier tranform of this substituted image, again XOR with chaotic sequence and obtained substituted image, finally Partial Fourier Transform and XOR with third chaotic sequence and ciphered image is obtained.

2.5 A Novel ⊗perator: In paper[9] Chiranjeev et al. proposed a new operator which can be used instead of traditional xor and p-m. This operator has certain properties like closure, associativity, identity, inverse and commutativity which makes it suitable to use in the image encryption. They have also applied this operator on single channel black and white images encryption and generated a mathematical encryption algorithm. The operator with the algorithm have given the satisfactory results.

Different authors have used number of algortihms for color image's encryption. To improve the execution of the cryptosystem, they entails several operations including DNA based encoding, DNA-X⊕R, DNA complement, DNA decoding, etc. The majority of cryptosystems rely on addition–subtraction (+,−) operations [3] or X⊕R operations [2,3,4], each of which has certain drawbacks. The disadvantages of using XOR in these operation are proved in paper [9]. In the same paper to overcome these limitation a new binary operator is proposed which is applicable in both conventional and DNA based cryptosystems.

Chapter 3: The Proposed Cryptosystem

Overview

In this chapter, an overview of the cryptosystem is mentioned in detail. This chapter contains every detail about pre requisite of an encryption as well as decryption algorithm. The mathematical functions which are required for encryption and decryption are chaotic maps, initial array(vector), rule charts(maps) and DNA's substitution maps.

Organization of this chapter

Several sections this chapter inhibit are Section 3.1 covers the proposed work which describes the steps of encryption algorithm. Section 3.2 summarises about the private key and chaotic(dynamical systems) maps and the initial values required to generated them. Section 3.3 outlines the generation of initial array(vector), its algorithm and mathematical formulation. Section 3.4 dicusses about the rule charts(maps), its algorithm and mathematical definition. Section 3.5 explains the DNA substitution charts(maps) and its formation. After generation of these requisites we can move ahead for ecryption. These sections also present how these charts/maps looks like in a matrix form.

3.1 The proposed work

An encryption and decryption technique utilising an operator $\otimes$ defined in [9] is presented in the proposed work. Measurements of the cryptosystem's performance have also been made using the experimental results and analysis to make sure operator works well with color images. Lastly, a comparison with other standard operators, such as X⊕R and addition-subtraction, is carried out. Also comparison with modern and current cryptosystem are done to verify that our cryptosystem is at par with other established color imaging cryptosystems.

Encryption algorithm steps are as follows in brief

- The new methods used in the algorithm, such as inter channel mixing, mix rows, spiral mixing are straightforward but have the power to significantly reduce the visual information in the original images. One of these steps i.e. mixrows requires to be operated with initial vector whose generation is described ahead.
- The cryptosystem utilizes the modified ACM algorithm proposed in paper [9], only images with similar dimensions can be shuffled using traditional ACM. Unevenly sized images shuffling is also possible with the modified ACM.
- To improve security and give the encryption process non-linearity, the suggested cryptosystem makes use of DNA[A,C,T,G] encoding, substitution based on DNA with the proposed operator and DNA[A,C,T,G] decoding. The process of encodng, decoding requires rule charts(maps) and substitution requires rule charts as well as DS maps.
- The cryptosystem has been tested using a number of statistical and security measures, together with experimentation and analysis to improve its performance, to ensure that it is resistant to a variety of threats.
- The comparative analysis shows that suggested operator outperforms other operators, such as XOR or (+,−) with the color images as well, and hence overcomes their shortcomings. Comparing our results with the result of other cryptosystem represent the accurate performance of our cryptosystem.

3.2 Private key and Chaotic Map

Private key is a critical component of many encryption systems, particularly in asymmetric cryptography, where it is used in conjunction with a corresponding public key. The private key is kept secret and is used primarily for decrypting data that has been encrypted with the public key or for digitally signing messages to verify the identity of the sender. In our cryptosystem symmetric key is used to generate the initial values of chaotic map which is an essential part of the process.

In our cryptosystem secret key is used to generate the initial conditions for the first chaotic system. It Is divided in 7 parts (p, q, x, y, t, cone, ctwo) and is of 240 bit. These seven parts are lengths of 40 bits, 40 bits, 40 bits, 40 bits, 40 bits, 20 bits and 20 bits respectively. The structure of the key is as folllows:

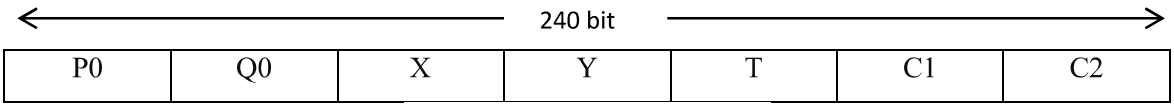


Fig 3.1 Secret Key Structure

The values of p,q, x, y, t, c1, c2 are generated as described in [7]

$$P0 = \sum_{i=0}^{39} \frac{key(i) * 2^{40-i}}{2^{40}} \qquad Q0 = \sum_{i=40}^{79} \frac{key(i) * 2^{80-i}}{2^{40}} \qquad X = \sum_{i=80}^{119} \frac{key(i) * 2^{120-i}}{2^{40}}$$

$$Y = \sum_{i=120}^{159} \frac{key(i) * 2^{160-i}}{2^{40}} \qquad T = \sum_{i=160}^{199} \frac{key(i) * 2^{200-i}}{2^{40}} \qquad C1 = \sum_{i=200}^{219} \frac{key(i) * 2^{220-i}}{2^{20}}$$

$$C2 = \sum_{i=220}^{239} \frac{key(i) * 2^{240-i}}{2^{20}}$$

After the generation of these values we proceed to generate the initial conditions of chaotic map. Chaotic map takes input of p, q, x0, y0 and gives output as x1, y1. The values of p, q, x0 and y0 are generated using these equations as given in paper[7]

$$p = (P0 + T * c1)\%5 + 16$$
$$q = (Q0 + T * C2)\%5 + 16$$
$$x0 = (X + T * C1)\%2 - 1$$
$$y0 = (Y + T * C2)\%2 - 1$$

Knowing these values we can proceed to chaotic Map

Chaotic Map(CM), often referred to as dynamical systems or nonlinear systems, are mathematical models that exhibit chaotic behavior. Unlike linear systems, where small changes in initial conditions lead to proportional changes in outcomes, chaotic maps demonstrate sensitivity to initial conditions, with slight variations resulting in vastly different trajectories over time. This property, known as sensitivity to initial conditions or the "butterfly effect," means that even minute differences in starting conditions can lead to significant divergence in system behavior.

Chaotic maps are characterized by their deterministic nature, meaning that their future states are entirely determined by their current state and the governing equations, without any randomness involved. Despite their deterministic nature, chaotic maps produce

seemingly random and unpredictable outcomes, making them valuable in various applications such as cryptography, data encryption, and secure communication protocols.

In our suggested cryptosystem we are using two dimensional multiple collapse chaotic map(2D MCCM) which has been thoroughly covered in the paper[11], uses a symmetric key with a length of 240 bits. These equations are used to construct the MCCM map.

$$Xn+1 = \arctan\left(\frac{q}{10 * p * yn} + \tan(p * \pi * xn)\right)$$

$$Yn+1 = \arctan\left(\frac{q}{p * xn} + \tan(10 * p * \pi * yn)\right)$$

where the values of the tuning parameters, p and q, are taken from the key. The starting value of xn and yn are also taken from the key as x0 and y0. To expand the map from 0 to any number n, the values of x_i and y_i are updated from the previous values where i=1 to n.

Demonstration of creating MCCM map from the initial values of size m*2 where m is the dimension of an image of size m*n using the given equations:

Defining twoDMCCM function

Input: a, b, x0, y0

```
function twoDMCCM(a, b, x0, y0):
    x1 = math.atan((b / (10 * a * y)) + math.tan(a * math.pi * x))
    y1 = math.atan((b / (a * x)) + math.tan(10 *a * math.pi * y ))
    return x1,y1
```

Output: x1,y1

Defining a function to create chaotic map

Input: a, b, x0, y0, m

```
Function getChaoticMap(a, b, x0, y0, m):
        Map=[]
        for i in range(self.m):
                x1,y1 = twoDMCCM(a,b,x0,y0)
                Map.append(list([x1,y1]))
                x0=x1, y0=y1
        return Map
```

Output: A m*2 dimensionsal array having values from 2DMCCM

3.3 Initial vector

With the first element of each row of the given image, the initial array (InV) is utilised to generate a quasirandom sequence that may be operated with $\otimes$. For an image of size m * n and bit depth P, the IV-dimension is 1 × m. The values are obtained in the following way from C$M(1)$.

$$\mathrm{InV}\,(1,\mathrm{i}) = |\mathrm{xi}| \times |\mathrm{yi}| \times (2^{31} - 1)\mathrm{mod}\,(2^{\mathrm{p}})$$
$$xi\,, yi \in M(1), i = 1, 2, \ldots m$$

The below function can be utilised to generate initial array(vector)

Defining function for initial array

Input: CM,m, pixel depth pd

```
Function getInitialVector(map,pd, m):
    IV = []
    for i starting from 0 to m:
        x1,y1 = map[i]
        InV = abs(x1) * abs(y1) * ((pow(2,31)-1) % pow(2,pd))
        IV.append(InV)
        return IV
```

Output: Initial array of size 1*m

3.4 Rule Maps

To construct one of the eight DNA encoding-decoding principle, as listed in Table 3.1, the rule mappings are utilised to create a quasirandom sequence of integers from 1 - 8. Because every two bits are mapped to one nucleotide, an image of size m * n of bit depth p will have a rule map of dimension $m \times (n \cdot (p{*}2))$. The dynamical map is sclaed to the dimension of image in order to produce the rule map. Modular operation is used to transform the precept, which are preferably selected from each dimension, to values between 1 and 8.

Getting Map of rules: To obtain this map, the chaotic map CM2 is expanded to the size same as image, the rules are choosen using some operations and are stored in the map. The map contains four rules for every pixel value as one pixel value is of 8 bit and one rule is for 2 bit that means for rules for 8 bit value.

The block diagram shows how the rule mappigs are obtained

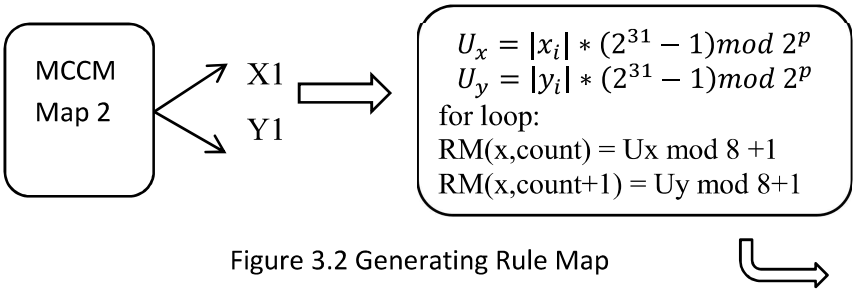


Figure 3.2 Generating Rule Map

	0	1	2	3
0	2	6	4	2
1	8	6	3	3
2	5	6	2	8
3	1	3	1	4
4	3	4	7	2
...	...	...	...	...
262139	5	7	6	1
262140	6	2	4	5
262141	7	8	2	3
262142	3	1	5	3
262143	2	8	7	3

262144 rows × 4 columns

Table 3.1
DNA Encoding rules

Binary digits	R1	R2	R3	R4	R5	R6	R7	R8
00	A	A	C	G	C	G	T	T
01	C	G	A	A	T	T	C	G
10	G	C	T	T	A	A	G	C
11	T	T	G	C	G	C	A	A

Below is the definition of a function that can generate rule map using the equations

Defining Rule Map function

Input: CM2, pixel depth pd, m, n

```
RM = []
for i starting from 0 to m:
        counter=1
        for j starting from 0 to n:
                Xplace,Yplace = CM[i]
                Ux = abs(Xplace) * ((pow(2,31)-1) % pow(2,pd))
                Uy = abs(Yplace) * ((pow(2,31)-1) % pow(2,pd))
                lit=[]
                for k starting from 0 to p/4:
                        lit.insert(counter,int(Ux % 8 +1))
                        lit.insert(counter+1,int(Uy % 8 + 1))
                        counter = counter + 2;
                        Ux = math.floor(Ux/8)
                        Uy = math.floor(Uy/8)
                RM.append(lit)
```

Output: Rules mapping contained in an array of m*(n*(p/2))* 4 dimension

The rule mappings are used with encoding rules table to map binary digit to corresponding DNA nucleotides.

3.5 Substitution Map for DNAs

Using the proposed operator ⊗, the DNA Substitution chart—a pseudorandom string of DNA nucleotides (A, C, T, G)—is utilised to work with the mixed image which is DNA encoded. With size of m*n an image with a p bits pixel depth will be enlarged to a DNA sequence of size $m \times (n.(p/2))$ which the DNA Substitution Chart's size ought to correspond to. The instruction for constructing a DNA Substitution chart is described in paper[9].

When the mixed image is encoded with DNAs, these DNAs then are operated with corresponding DNAs from DSMap according the rule decided by the rule map. This DSMap is required at the later stages in encryption and earlier stages in decryption.

Getting substitution map for DNAs: For getting DSM, the chaotic map 3 is elaborated till the size of image, for every pixel there will be four DNA nucleotide present in the map as every two bit of the pixel will be converted to one of the nucleotide. The dimension of the DSM will be same as of RM.

Below is the block diagram to construct the DSMap

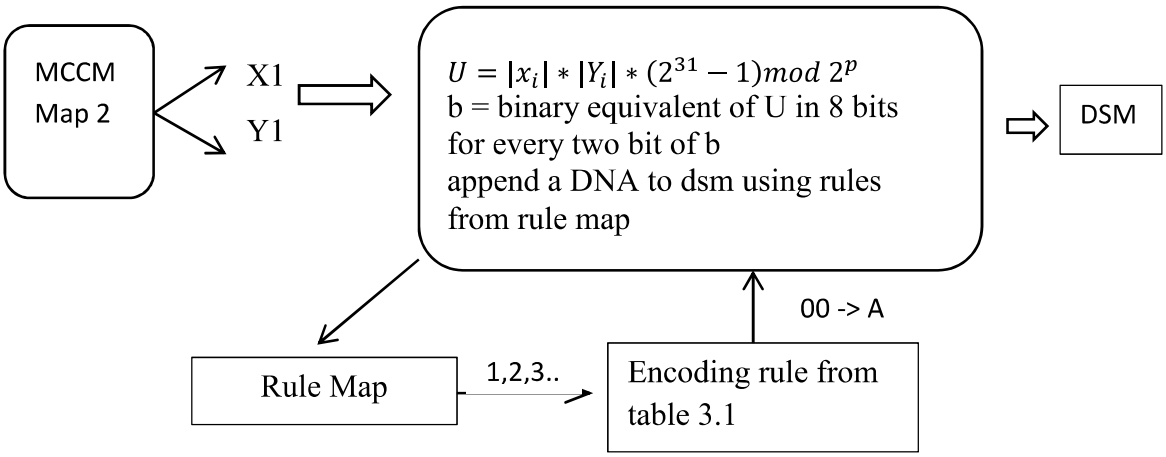


Figure 3.3 Generating DS Map

The generated Dna Substitution map are very usefull in encryption process as DNAs introduces non-linearity to the cryptosystem and then substituting them with other DNAs make this system highly protective as the original DSNs are now hidden as the form of substituted DNAs then these only are decoded as a cipher.

	0	1	2	3
0	C	G	A	G
1	A	A	T	A
2	G	T	A	T
3	G	C	A	T
4	T	T	A	A
...	...	...	...	...
262139	G	G	A	C
262140	C	A	A	A
262141	G	G	A	T
262142	G	G	A	A
262143	C	A	C	A

262144 rows × 4 columns

Fig 3.4 Generated DSM

Below is the function to construct DSM

Defining of function for DSM

Input: RM, CM3, pixel depth pd, m, n

```
function getDSMap(RM, CM, pd):
        DSM = []
        for i starting from 0 to m:
            counter = 0
            for j starting from 0 to n:
                x,y = CM[i]
                U = abs(x) * abs(y) * ((pow(2,31)-1) % pow(2,pd))
                bVal = binary equivalent of U in p bit
                lst=[]
                for k starting from 0 to length of bval with 2 jumps:
                    str=bVal[k]+bVal[k+1]
                    lit = EncodingRules.get(str)
                    DSM.append(lit[RM[((i-1)*self.n+(j-1))][count%4]-1])
                    count = count + 1
        return DSM
```

Output: DS map containing nucleotide for every two bit per pixel

After obtaining the necessary maps, the images can be encrypted. The decryption procedure also requires the same maps. The next chapters provides a description of the encryption and decryption procedure respectively.

Chapter 4: Encryption

Overview

In this chapter, we delve into the intricate world of encryption, exploring its fundamental principles and components. The chapter provides a compreensive details of our encryption scheme and the steps that are the part of encryption procedure.

Organization of the chapter

This chapter is the most important as the real work lies here. This chapter is divided in eight sections. Section 4.1 starts with intoduction to the encryption procedure in brief and it contain the most useful block diagram. After this section every section itself is one of the step of encryption algorithm. Section 4.2 outlines the first step i.e. splitting color image into three channels viz, red, green and blue. Section 4.3 summarises the second step i.e. inter channel mixing. Section 4.4 covers third step named as mix rows. Section 4.5 discusses the next step called as ACM. Section 4.6 explains the fifth step know as spiral mixing. Section 4.7 clarify the encoding process followed by section 4.8 containing DNA subtitution. The last section 4.9 clear up the decoding process and finally merging decoded images constitute to final cipher image.

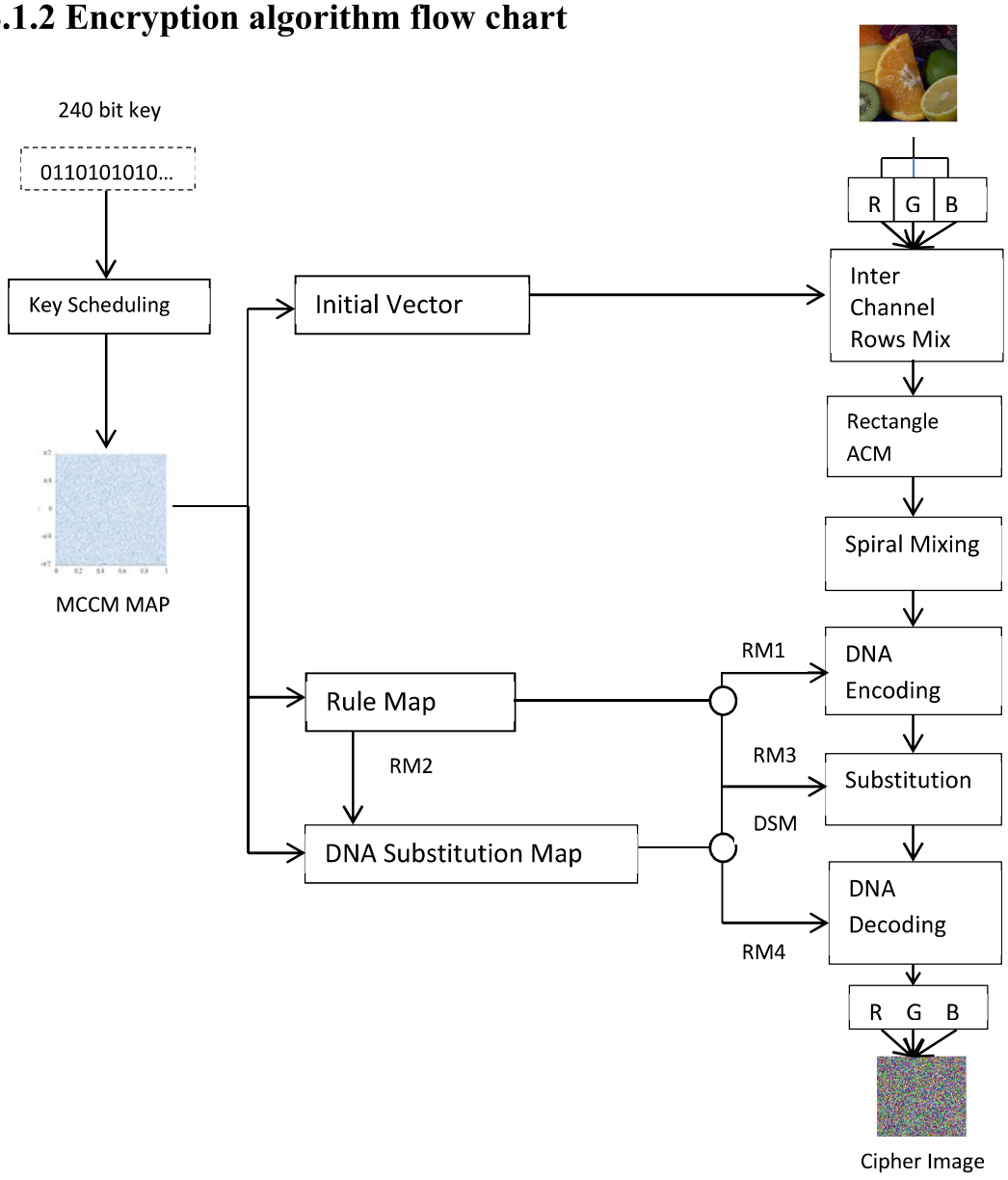
4.1.1 Introduction

The encryption procedure makes use of six 2D-Multiple Collapse Chaotic Maps, which are produced by the Key Scheduling Algorithm from a 240-bit symmetric key. The nonlinear maps produce quasirandom sequences in the interval [−1, 1], from which certain functions are used to construct values and maps such as DNA substitution maps, rule maps, and initial vector. The different procedures required to encrypt the image make use of these variables and mappings.

The first stage involves intermixing of channels of the image then utilising the proposed operator $\otimes$ to operate the rows of the intermixed images with the initial array. Using a modified version of Arnold's Cat Map [10], which can be applied to both square and rectangular images, the pixels of the composite image are jumbled. The suggested

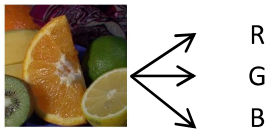
operator is then used to further mix the pixels in the directions of forward spiral row, forward spiral column, reverse spiral row, and reverse spiral column. These keyless procedures are necessary to eliminate the link between pixels and the image's visual perception. Lastly, a non-linear layer has been added by combining the suggested operator with the idea of DNA cryptography. One of the eight encoding principle identified by the rule chart is applied to the mixed image that was obtained in the preceding phases, converting it into a sequence of DNA nucleotides. At this stage DNA encoded image is obtained and these encodings are further substituted using proposed operator with the substitution map. Lastly, one of the eight DNA decoding guidelines identified by a different rule chart is applied to decode the substituted DNA sequences.

4.1.2 Encryption algorithm flow chart



4.2 Image Loading and Splitting

Split the R, G, B pixel matrices from the color image into three different matrix say red, green, blue. We need to apply all steps to every matrix inorder to encrypt the image in best way.



4.3 Inter Channel Mixing

In this stage, the channels of the image are intermixied with each other using some equations. The splitted channels viz. R, G, B are intermixed together using some equations. By blending information from different color channels inter-channel mixing introduces complexity and randomness into the encryption process, making it more resistant to various cryptographic attacks. Additionally, this technique helps to distribute the encrypted data across multiple channels, thereby reducing the vulnerability to targeted attacks on specific color components. Overall, inter-channel mixing contributes to the overall strength and effectiveness of the image encryption algorithm by increasing its entropy and making it harder for adversaries to decrypt the protected images. The equations for inter channel mixing are as follows

$$R' = R \otimes (G \otimes B)$$
$$G' = G \otimes (R' \otimes B)$$
$$B' = B \otimes (R' \otimes B')$$

where $\otimes$ is the operator defined in paper[9], R', G', B' are inter channel mixed componenets. Below is the block diagram illustartion of inter channel mixing.

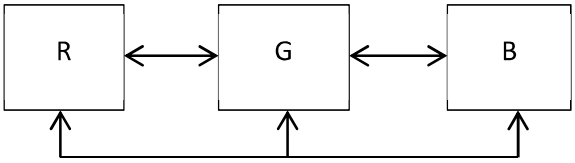


Fig 4.2 Inter Channel Mixing

Below is the definition of a function for inter channel mixing.

Defining function for inter channel mixing

Input: r, g, b

function interMix(r, g, b):

m= length of r

n= length of first row

imix = [][]

brac = g will operate with b using ⊗

Imix = r will operate with brac using ⊗

return imix

Output: inter channel mixed image

The process of inter channel mixrows will be applied on the image received after splitting color image into matrices i.e. R, G, B, respectively as.

$$R' = interMix(r, g, b)$$
$$G' = interMix(g, R', b)$$
$$B' = interMix(b, R', G')$$

4.4 Mix Rows

The values from the initial vector (IV), which has already been generated, are mixed and distributed throughout the inter mixed image's rows. The IV will have a dimension of 1 * m for a plain image I with dimensions m * n. To create image I', the first value in each row of the matrix is operated with ⊗ using the appropriate values of the initial array. The updated value of the first element in each row is propagated along its neighbor pixels in the whole row using the following equation to skew the value in the neighbouring pixels:

$$I'(g, h) = I(g, h) \otimes I'(g, h - 1);$$
$$h = 2, 3 \ldots, n; \forall g \in \{1, \ldots, m\}$$

The equation likely involves a combination of mathematical operations tailored to disperse the encrypted information effectively across the entire row while maintaining a level of unpredictability. Overall, this operation of mix rows ensures that the encrypted

information from the IV is spread throughout the image rows in a manner that enhances security and prevents easy decryption. Below is the illustartion of mix rows operation.

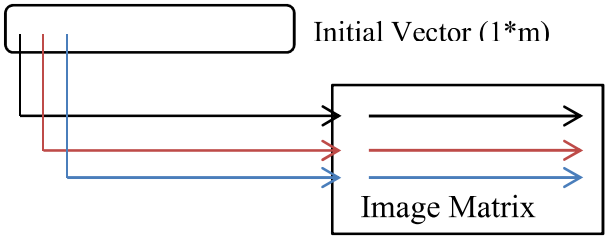


Fig 4.3 Mix rows

Definition of MixRows function

Input: Inter channel mixed image, InV

```
function mixRows(I', IV):
    for i starting from 0 to size of image
        mixedI[i][0] = apply operator on I'[i][0], IV[i]
        for j starting from 1 to length of row
            mixedI[i][j] = apply operator on I'[i][j], mixedI[i][j-1])
    return mixedI
```

Output: mixed rows image

The process of mix rows will be applied on inter channel mixed image as

$$mixedImageRed = mixRows(R', IV)$$
$$mixedImageGreen = mixRows(G', IV)$$
$$mixedImageBlue = mixRows(B', IV)$$

4.5 Arnold Cat Map(ACM) algorithm

A: Arnold C: Cat M: Map(ACM)[10] is a disorganised chart that is used to rearrange the given image. When a 2D array has dimensions of N*N, ACM is expressed as:

$$\begin{matrix} g' \\ h' \end{matrix} = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix} \begin{matrix} g \\ h \end{matrix} (mod\ N)$$

where g, h, g′, h′ ∈ [1, N]. (g, h) and (g′, h′) represents, respectively, the original and scrambled images' position indices. The drawback of employing directly ACM algorithm is that its definition is limited to identical size images. Nonetheless, uneven proportions are a regular occurrence in both natural and medical imaging. The image is

separated into minimal squares, and ACM is applied independently to each of them in order to fix this problem.

Let m*n represent an image's dimension. To divide it into as few squares as feasible, the length of each square's side will be $N = \min(m, n)$. Therefore, the proportion of squares that are obtained will be $\alpha = \lceil \max(m, n)/N \rceil$. If m and n are distinct combination of one another, then $L = N - (\max(m, n) \bmod N)$), an additional length L pixel remains with the longer side. There will be some overlap between the squares in order to accommodate this additional space within the image. There can be a maximum of $\alpha - 1$ overlaps to ensure almost equal overlapping; the duration of the last overlap i.e. eta=$\lfloor L/(\alpha - 1) \rfloor$. The last overlap modifies the excess portion that is still present.

Given that $m < n$ for an image, $N = m$ and the initial value of I[1,1] of first square, a = 1, b = 1. For the initial $\alpha - 1$ squares, the subsequent values will be ascertained as $b' = b + N - \text{eta}$. To accommodate the extra square that remains, the y-coordinate for the last square will be $y-N$. For images when $m>n$, same computations can be performed by transposing the m and n values. Next, ACM is applied to each separated square individually. On the other hand, ACM is regular; that is, the original image is restored following a definite number of iterations. The image N's dimension determines this regularity. To achieve a well-shuffled image and avoid regularity, ACM will be applied to the image itself t number of times, where t is the highest prime number less than or equal to (N/2), N being the size of each image.

The process of applying ACM on images will be applied on the image received after mixrows operation on every matrix i.e. R, G, B, respectively as

$$imageACMR = applyACM(mixedImageRed)$$
$$imageACMG = applyACM(mixedImageGreen)$$
$$imageACMB = applyACM(mixedImageBlue)$$

The algorithm for rectangular ACM is defined in paper[9]. ACM algorithm is already defined, but due to its restrictions a modified algorithm is defined in this paper. The main benefit is that this algorithm does not require any shape effects or difference in size of image is not taken care of. If we pass uneven size image to traditional ACM it will fail.

4.6 Spiral Mixing

In order to prevent differential attacks on the image, this stage job is to propagate the pixel's value information throughout it in spiral manner. This procedure helps remove the image's visual information even if it doesn't require an encryption key. Let m and n represent the input image I's row and column counts, respectively.

The primary goal of this step is to diffuse the pixel values across the image to prevent recognizable patterns from being exploited in a differential attack. By rearranging pixel values in a non-linear and complex pattern like a spiral, the algorithm diminishes the visible clues that could potentially reveal the original content or the nature of the encryption algorithm. The algorithm has four steps

- **Forward Spiral Row Mixing:** The aim here is to propagate pixel values horizontally across the rows in a forward-moving spiral pattern. This step effectively scrambles the row data, making it harder to discern patterns that might reveal information about the original image.

Definition of function for this step

```
function spiralRow(M):
        for i starting from 0 to m:
                if i is even:
                        if i!=0: (edge case)
                                M[i][0] =M[i][0] ⊗ M [i-1][0]
                        for j starting from1 to n:
                                M[i][j] = M[i][j] ⊗ M[i][j-1]
                        M[i+1][j] =M [i+1][j] ⊗ M[i][j]
                else:
                        for j starting from n-2 to 0:
                                M[i][j] = M [i][j] ⊗ M[i][j+1]

        return M
```

- **Forward Spiral Column Mixing:** Similar to the spiral row mixing, but this time the pixel values are propagated vertically down the columns in a forward-moving spiral pattern. This step further obscures the vertical structures and patterns in the image.

Definition of function for this step

```
function spiralColumn(M):
    for each column starting from 0 to n-1:
        if j is even:
            If j ≠ 0: (edge case)
                M[0][j] = M[0][j] ⊗ M[0][j-1]
                For each row starting from 1 to m-1:
                    M[i][j] = M[i][j] ⊗ M[i-1][j]
                M[i][j+1] = M[i][j+1] ⊗ M[i-1][j]
            else:
                For each row starting from m-2 to 0:
                    M[i][j] = M[i][j] ⊗ M[i+1][j]
    return M
```

- **Backward Spiral Row Mixing:** To add an additional layer of security, pixels could now be mixed in a reverse spiral pattern across rows, essentially retracing the forward steps. Going from the end point to upward direction,also we can say bottom up traversal.

Defining function for this step

```
function bspiralRow(M):
    for i starting from m to 0:
        if i is even:
            for each column j starting  from 1 to n-1:
                M[i][j] = M[i][j] ⊗ M[i][j-1]
        else: (if i is odd)
            If i != m-1:
                M[i][n-1] = M[i][n-1] ⊗ M[i+1][n-1]
                For each column j from  n-1 to 1:
                    M[i][j-1] = M[i][j-1] ⊗ M[i][j]
                M[i-1][0] = M[i-1][0] ⊗ M[i][0]
    return M
```

- **Backward Spiral Column Mixing:** Finally, a reverse movement in the columns ensures that any vertical patterns disrupted by forward column mixing are further encrypted. Going from the last point towards first in column direction. Also it can be called bottom up traversal. These processes just make the image more mixed and pixel propagated.

Definition of function for this step

```
function bspiralColumn(M):
        For each column j starting from n-1 to 0:
                If j is even:
                        For each row i starting from 1 to m-1:
                                M[i][j] = M[i][j] ⊗ M[i-1][j]
                Else (if j is odd):
                        If j ≠ n-1:
                                M[i][j] = M[i][j] ⊗ M[i][j+1]
                        For each row i from m-1 to 1:
                                M[i-1][j] = M[i-1][j] ⊗ M[i][j]
                        M[i-1][j-1] = M[i-1][j-1] ⊗ M[i-1][j]
        return M
```

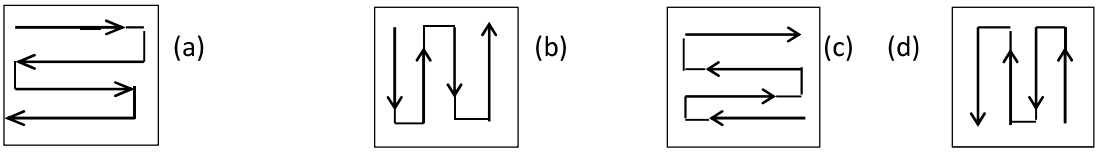


Fig 4.4: a – d are the spiral mixing operations described ahead.

Defining Spiral Mixing function

Input: Matrix after ACM

```
function spiralMixing(M):
        M = spiralRow(M)
        M = spiralColumn(M)
        M = bspiralRow(M)
        M = bspiralColumn(M)
        retrurn M
```

Output: Spiral mixed image

In summary, this stage of the image encryption process using spiral diffusion is critical for enhancing security by effectively removing recognizable visual information and making the image resistant to analysis and decryption attempts through differential cryptanalysis. This method enhances the robustness of the encryption without relying on traditional cryptographic keys.

4.7 DNA based Encoding

One of the encoding principle transforms each pixel value in the image into a string of DNA nucleotides. At a time, two bits are extracted and transformed into a nucleotide by applying the rule found in the obtained rule map. As a result, a string half of length i.e. (p/2) nucleotides per pixel will be created from an image with a bit depth of *p*-bit.

Below is the block diagram shows how the encoding process executes

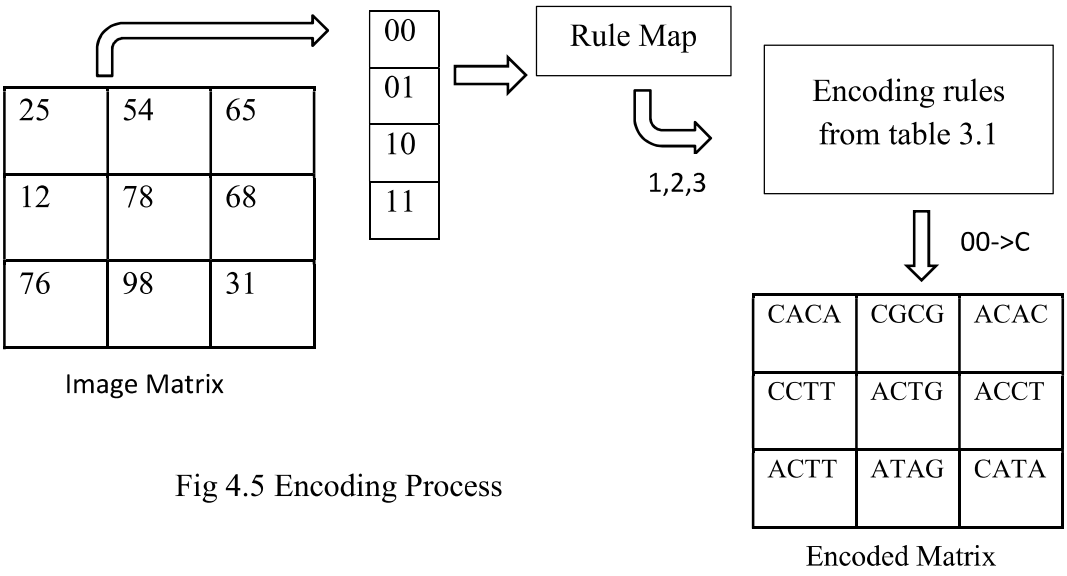


Fig 4.5 Encoding Process

Defining function for encoding process

Input: Image Matrix M, Rule Map, Encoding rules

```
function encoding(RM, M, encodingRules):
    for x starting from 0 to length of image:
        set counter = 0
            for y starting from 0 to length of row:
                bnry = binary of  M[x][y]
                create lst=[]
                for z starting from 0 to length of bnry taking two digits
                    str=bnry[z]+bnry[z+1]
                    list = encodingRules.get(str)
                    lst.append(list[(RM[(x*n+y)][count%4])-1])
                    set counter = counter + 1
                    encodedImage.append(lst)
    return encodedImage
```

Output: Encoded image

4.8 Substitution of DNAs

The DNA sequences D that are received after the image has been encoded are substituted in the replacement process with ⊗ using the sequence that is obtained from the DNA replacement Map (DSM). The mapping of $\Sigma = \{A, C, T, G\}$ to S = {0, 1, 2, 3} for DNA operation requires the selection of one among eight encoding rules from the values derived from the rule map (RM). Eight options will appear when ⊗ is used, one for each of the encoding rules that are applied to a nucleotide's pair; yet, some pairs may give identical outcomes for distinct rules. Since of this, the dynamical DNA sequence that is left over after this procedure is resistant to algebraic attack since it depends on the rule's table as well as the elements of the DNA substitution table, both of which are necessary to decrypt the cryptic image and determine the original key.

The following represents the outcome of DNA operations that correspond to the various encoding rules:

	A	C	G	T
A	A	C	G	T
C	C	T	A	G
G	G	A	T	C
T	T	G	C	A

R1

	A	C	G	T
A	A	C	G	T
C	C	T	A	G
G	G	A	T	C
T	T	G	C	A

R2

	A	C	G	T
A	G	A	T	C
C	A	C	G	T
G	T	G	C	A
T	C	T	A	G

R3

	A	C	G	T
A	C	T	A	G
C	T	G	V	A
G	A	C	G	T
T	G	A	T	C

R4

	A	C	G	T
A	G	A	T	C
C	A	C	G	T
G	T	G	C	A
T	C	T	A	G

R5

	A	C	G	T
A	G	A	T	C
C	A	C	G	T
G	T	G	C	A
T	C	T	A	G

R6

	A	C	G	T
A	T	G	C	A
C	G	A	T	C
G	C	T	A	G
T	A	C	G	T

R7

	A	C	G	T
A	T	G	C	A
C	G	A	T	C
G	C	T	A	G
T	A	C	G	T

R8

Table 4.1 DNA rules for substitution

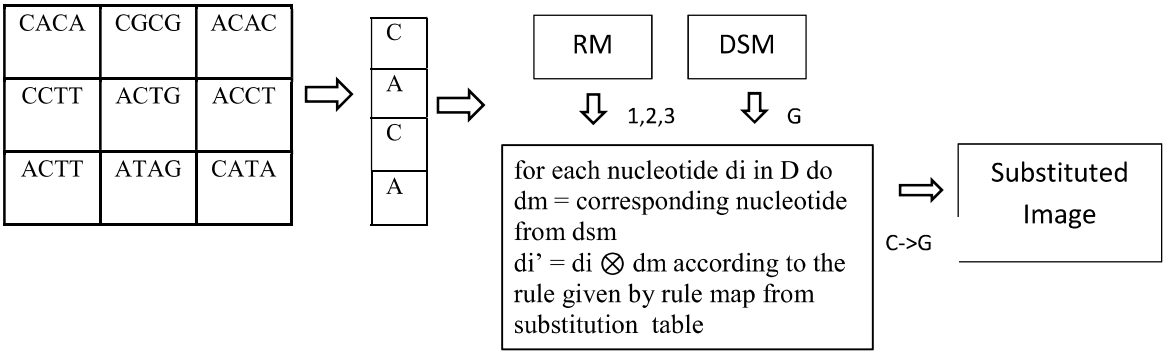


Fig 4.6 Substitution Process

Defining function for substitution process

Input: DNA encoded image eM, RM, DSM

```
function substitution(eM, RM, DSM):
        for i starting from 0 to length of eM:
                lst=[]
                for j starting from 0 to length of row:
                        dM = corresponding DNA nucleotide from DSM
                        rule = corresponding rule from RM
                        dI = getRule(rule,dM,encodedImage[i][j]) from substitution table
                        lst.append(dI)
                        substituted.append(lst)
        return substituted
```

Output: substituted image

4.9 Decoding of DNAs

One of the eight decoding rules is used to the chaotic DNA sequences in order to transform them to decimal values; the rule map's sequences determine which rule is applied. Then, a single cipher color image is created by combining the three matrices—cipher red, cipher green, and cipher blue obtained after decoding. The outcome of this decoding process is the final cipher image. Below is decoding process.

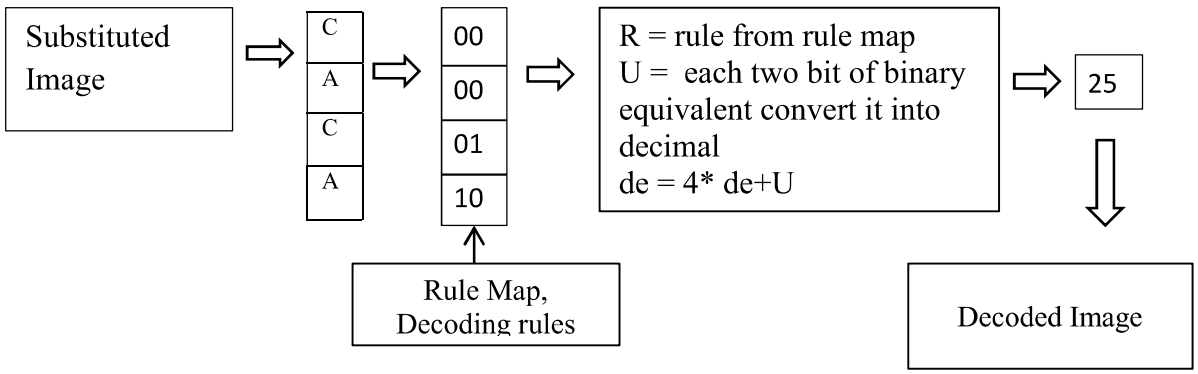


Fig 4.7 Decoding process

Table 4.2 DNA Decoding rules

DNA	R1	R2	R3	R4	R5	R6	R7	R8
A	00	00	01	01	10	10	11	11
C	01	10	00	11	00	11	01	10
G	10	01	11	00	11	00	10	01
T	11	11	10	10	01	01	00	00

Defining function for Decoding process

Input: Substituted Image sM, RM

```
function decoding(sM, RM):
    for i starting from 0 to length of sM:
        de = 0
        for j from 0 to 4:
            dD = sM[i][j]
            rules = RM[i][j]
            u = get binary equivalent(rules,dD)
            u = get decimal equivalent(u)
            de = 4*de+u
            Pfinal.append(de)
    return Pfinal
```

Output: Decoded Image

After spiral mixing we have to apply DNA operation on the mixed image as shown below.

Defining function for all DNA operations

Input: Image after spiral mixing, RM, DSM

```
function dnaOperations(image, RM, DSM):
    encImage = encoding(image, RM)
    subImage = substitution(encImage, RM, DSM)
    decImage = decoding(subImage, RM)
    return decImage
```

Output: DNA operated image

DNA operation will be applied on all the three components and then we can merge them for getting final cipher image as shown in encryption flow chart. The work on sender ends at encryption. Now image is ready to be send. The process of decryption takes place at reciever's end.

Chapter 5: DECRYPTION

Overview

In this chapter, we delve into the intricate world of decryption, unraveling its fundamental principles and components. The chapter provides comprehensive details of our decryption scheme and the steps that are part of the decryption procedure.

Organization of the chapter

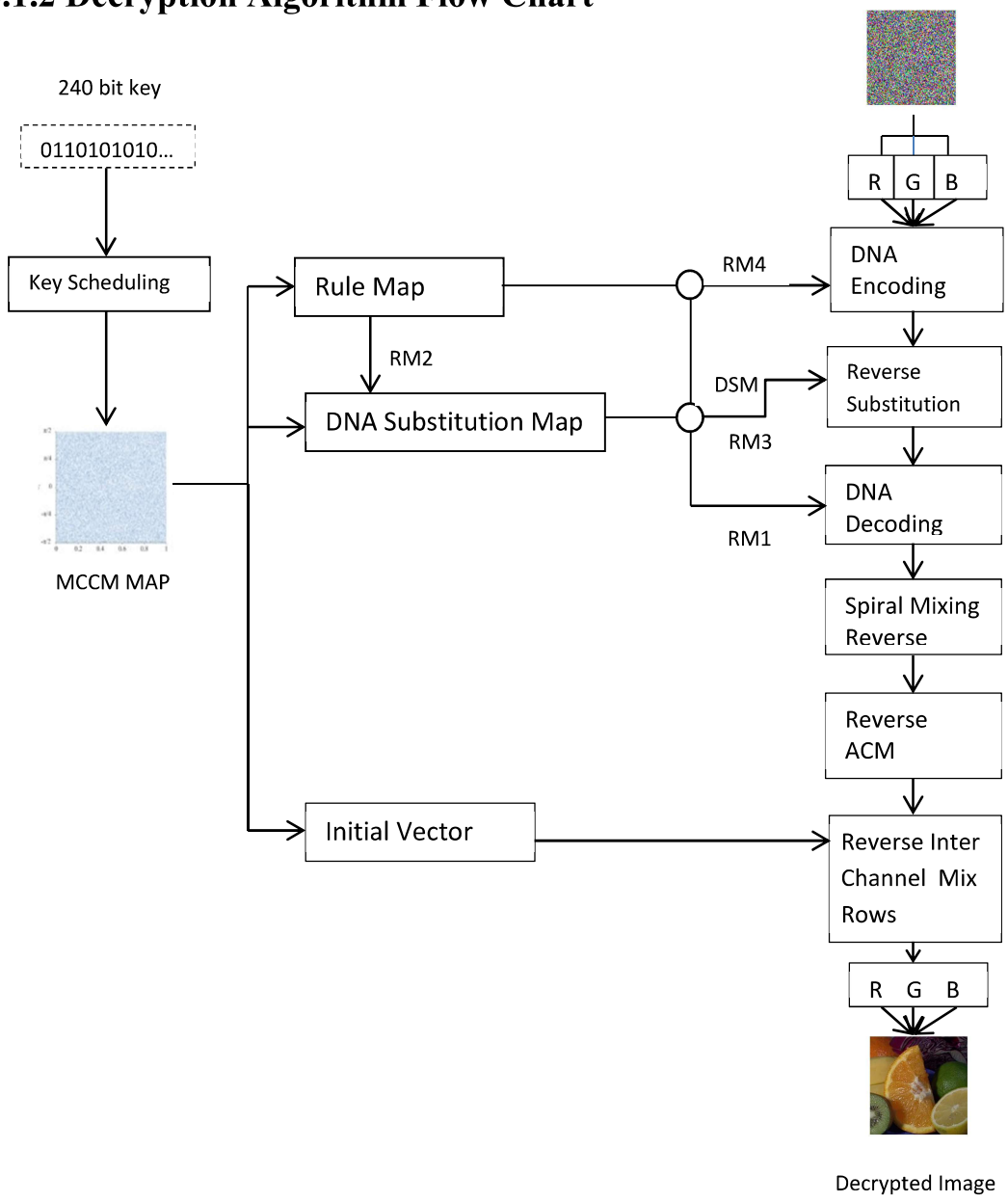
This chapter is divided in ten sections. Section 5.1 starts with intoduction to the decryption procedure in brief and it contain the most useful block diagram. After this section every section itself is one of the step of decryption algorithm. Section 5.2 outlines the first step i.e. splitting encrypted image into three channels viz, dred, dgreen and dblue. Section 5.3 summarises the second step i.e. DNA encoding. Section 5.4 covers third step named as reverse substitution. Section 5.5 discusses the next step called as DNA decoding. Section 5.6 defines reverse operator. Section 5.7 explains the fifth step know as spiral mixing. Section 5.8 clarify the reverse ACM process followed by section 5.9 containing reverse mix rows. The last section 5.10 clear up the reverse inter channel mix process and finally merging resultant images constitute to decrypted image.

5.1.1 Introduction

After obtaining cipher image we proceed for its decryption as on reciever side it needs to be decrypted. The encryption and decryption processes are inverted. The steps that are taken in last stages will now be taken in initial stages as we move in reverse fashion. As presented in flow chart, we again split the cipher images into R, G, B components and applies DNA encoding. If we compare encyption and decryption flow chart we can see we are dealing with DNAs in the later stages in encryption but in decryption we are using DNAs in initial stages. At first we do DNA encoding then we apply reverse DNA substitution using reverse substitution rules followed by DNA decoding. Encoding and Decoding have the same algorithm but substitution have minor change, it uses reverse

substitution rules. After decoding we go for reverse spiral mixing which internally follows reverse fashion of mixing in encryption. Then we apply reverse modified ACM followed by reverse mix rows and reverse inter channel mixing. At last we again merge RGB into final decrypted image. The main point of decryption is to use the same key as it was used in encryption for getting 2D MCCM, initial vector, rules maps and DNA substitution map.

5.1.2 Decryption Algorithm Flow Chart



The process of decryption will be successful only if right maps are used with right steps, as the rule maps are also changed with the steps.

5.2 Splitting Encrypted Images

In the decryption process, after the encrypted image has been retrieved, it's often necessary to split it back into its constituent color components—Red (R), Green (G), and Blue (B)—especially in the context of color images represented in the RGB color model. This step ensures that decryption can be applied to each color channel individually, allowing for the reconstruction of the original color image.

5.3 DNA Encoding

As a first step now DNA encoding is performed on the R, G, B, matrices received after splitting. The process of encoding is same in encryption as well as decryption. Also the function defined in encryption can also be used in decryption.

5.4 Reverse DNA Substitution

Reverse DNA substitution in decryption involves the reversal of the substitution process to retrieve the original encoded image from the DNA-like images. This reversal is typically achieved by applying an substitution rule with reverse algorithm that maps each DNA base pair or sequence back to its corresponding plaintext character or symbol. Implementing reverse DNA substitution decryption algorithms may be complex, requiring careful design and optimization to ensure efficiency and accuracy.

We drive reverse subtitution rules as given below

	A	C	G	T
A	A	G	C	T
C	C	A	T	G
G	G	T	A	C
T	T	C	G	A

R1

	A	C	G	T
A	A	G	C	T
C	C	A	T	G
G	G	T	A	C
T	T	C	G	A

R2

	A	C	G	T
A	C	A	T	G
C	T	C	G	A
G	A	G	C	T
T	G	T	A	C

R3

	A	C	G	T
A	G	T	A	C
C	A	G	V	T
G	G	C	G	A
T	C	A	T	G

R4

	A	C	G	T
A	C	A	T	G
C	T	C	G	A
G	A	G	C	T
T	G	T	A	C

R5

	A	C	G	T
A	G	T	A	C
C	A	G	C	T
G	T	C	G	A
T	C	A	T	G

R6

	A	C	G	T
A	A	G	C	T
C	G	T	A	C
G	C	A	T	G
T	A	G	C	T

R7

	A	C	G	T
A	A	G	C	T
C	C	A	T	G
G	C	A	T	G
T	A	G	C	T

R8

Table 5.1 Reverse DNA substitution rules

Defining Reverse substitution function

Input: DNA encoded matrix of cipher image

```
function reverseSubstitution(image):
        substituted = [ ]
        for i starting from length of encodedImage to 0:
                lst = [ ]
                for j starting from length of encodedImage to 0:
                        dM = DSM [ ( I * len(encodedImage[0]) ) + j ]
                        rule = RM [i][j]
                        dI = getRevRule(rule, dM, encodedImage[i][j])
                        lst.append(dI)
                substituted.append( lst )
```

Output: reverse substituted image

5.5 DNA Decoding

DNA decoding is the same process as decoding DNA nucleotide into decimal values as they make up the pixel matrix. The decoding process and algorithm is same as it is describes in encryption process.

5.6 Reverse Operator ($\otimes^{R}$)

For further steps of decryption we reverse the operator $\otimes$ in decryption algorithm as reverse operator which satisfy every condition and property of operator defined in paper[9].

$$(a \otimes^{R} b) = \big((a+1) * modinv(b+1, p)\big) mod\ p - 1$$

where we do first mod inverse of b+1 and pixel depth p then further calculation. The reverse operator $\otimes$^R is designed to reverse the effects of the original operator $\otimes$ used during encryption. It ensures that when applied to the cipher image, it effectively undoes the mixing applied during encryption, resulting in the recovery of the original plain images values. In summary, the reverse operator $\otimes^{R}$ plays a crucial role in the decryption algorithm by enabling the reversal of the effects of the original operator $\otimes$ used during encryption.

5.7 Reverse Spiral Mixing

In reverse spiral mixing, the encryption process is essentially reversed to reconstruct the original image from its encrypted form. This technique involves applying mixing operations in a reverse fashion. There will be same four operations but their line of operation will be reversed. Among the four the last one in encryption will be the first one in decryption.

- **Backward Spiral Column Mixing:** This operation is applied in column from last element towards first.

Defining function for this step
function rbsc(M): for each column j from 0 to n-1: if j is even for each row i from m-1 to 1 M[i][j] = M[i][j] $\otimes^R$ M[i-1][j] If j < n-1: M[i-1][j] = M[i-1][j] $\otimes^R$ M[i-1][j+1] else (if j is odd) for each row i from 0 to m-2 M[i][j] = M[i][j] $\otimes^R$ M[i+1][j] if j < n-2, M[i+1][j] = M[i+1][j] $\otimes^R$ M[i+1][j+1] return M

- **Backward Spiral Row Mixing:** Operation appliedon rows in the backward direction from last element towards first row wise.

Defining function for this step
function rbsr(M): for each row i from 0 to m-1: if i is even: for each column j from n-1 to 1: M[i][j] = M[i][j] $\otimes^R$ M[i][j-1] if i<m-1: M[i][j-1] = M[i][j-1] $\otimes^R$ M[i+1][j-1] else (if i is odd) for each column j from 0 to n-2 M[i][j] = M[i][j] $\otimes^R$ M[i][j+1] if i < m-1: M[i][j+1] = M[i][j+1] $\otimes^R$ M[i+1][j+1] return M

- **Forward Spiral Column Mixing:** Mixing in forward direction column wise

Defining function for thi step

```
function rfsc(M):
        for each column j from n-1 to 0:
                if j is even:
                        for each row i from m-1 to 1:
                                M[i][j] =M[i][j] ⊗^R M[i-1][j]
                        if j > 0, M[i-1][j] = M[i-1][j] ⊗^R M[i-1][j-1]
                else (if j is odd):
                        for each row i from 0 to m-2:
                                M[i][j] = M[i][j ⊗^R M[i+1][j]
                        if j > 0, M[i+1][j] =M[i+1][j] ⊗^R M[i+1][j-1]
        return M
```

- **Forward Spiral Row Mixing:** Row wise Mixing in forward direction

Defining function for this step

```
Function rfsr(M):
        for each row i from m-1 to 0:
                if i is even
                        for each column j from n-1 to 1
                                M[i][j] = M[i][j] ⊗^R M[i][j-1],
                        if i ≠ 0, M[i][j-1] = M[i][j-1] ⊗^R M[i-1][j-1],
                else (if i is odd):
                        for each column j from 0 to n-2
                                M[i][j] = M[i][j] ⊗^R M[i][j+1]
                        if i ≠ 0: M[i][j+1] = M[i][j+1] ⊗^R M[i-1][j+1]
        return M
```

All these function constitute to reverse spiral mixing as

Defining function for reverse spiral mixing

Input: DNA decoded image M

```
function revSpiralMixing(M):
        revbCol = rbsc(M)
        revbRow = rbsr(revbCol)
        revfCol = rfsc(revbRow)
        spiralMixed = rfsr(revfCol)
        return spiralMixed
```

Output: Reversely spiral mixed image

5.8 Reverse ACM

ACM is shuffling algorithm which shuffles the image pixelto different locations. Reverse ACM will shuffle back the pixel to their original place. In reverse ACM algorithm we just required to reverse the loops. The ACM algorithm is given paper[9] just reverse the loop and the desired output will be obtained.

5.9 Reverse Mix Rows

In the decryption process, after applying reverse modified ACM (Arnold Cat Map) to the encrypted image, the next step involves treating the rows of the resulting image with the same Initialization Vector (IV) used during encryption and the reverse of the operator denoted as $\otimes^{R}$

Defining function for Reverse mix rows

Input: Reversed ACM image

```
function revMixRows(image):
    for s starting from m-1 to 0:
        for t starting from n-1 to 1:
            Dimg[s][t] = image[s][t] ⊗^R image[s][t-1]
        Dimg[s][0] = image[s][0] ⊗^R IV[t]
    return Dimg
```

Output: Reversely mixed row image

5.10 Reverse Inter Channel Mixing

After applying mix rows, the channel are reversly diversed to get the original r, g, b matrices. We defined inter mix equations in such a way that they can be reversed easily. Below equations shows the reverse procedure. For reverse intermix we start with getting B channel.

$$B = B' \otimes^{R} (R' \otimes G')$$
$$G = G' \otimes^{R} (R' \otimes B)$$
$$R = R' \otimes^{R} (G \otimes B)$$

After applying these function we can get back the original R, G, B matrices without a single pixel difference.

Defining function for reverse inter channel mixing

Input: Reversed mixed row image

function revInterMix(r, g, b):

m= length of r

n= length of first row

imixr = [][]

brack = g will operate with b using $\otimes$

imixr = r will operate with brack using $\otimes^{R}$

return imixr

Output: Original Image

Finally Merge the decrypted R, G, B matrices to get final decrypted image.

Chapter 6: Experimental Results and Analysis

Overview

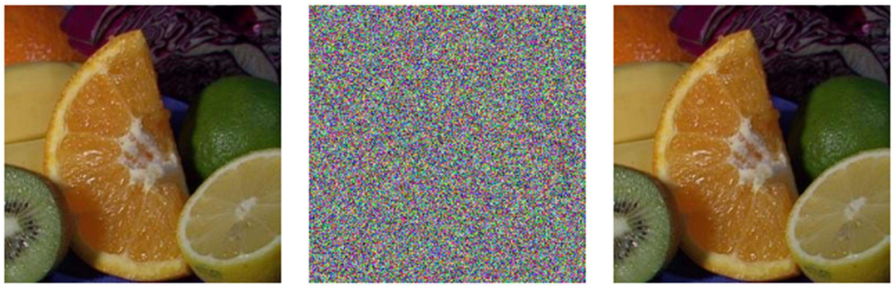
The chapter on experimental results and analysis provides a comprehensive examination of the outcomes obtained from applying the encryption and decryption algorithms to various datasets. It involves the presentation, interpretation, and discussion of quantitative and qualitative findings derived from experimentation.

Organization of the chapter

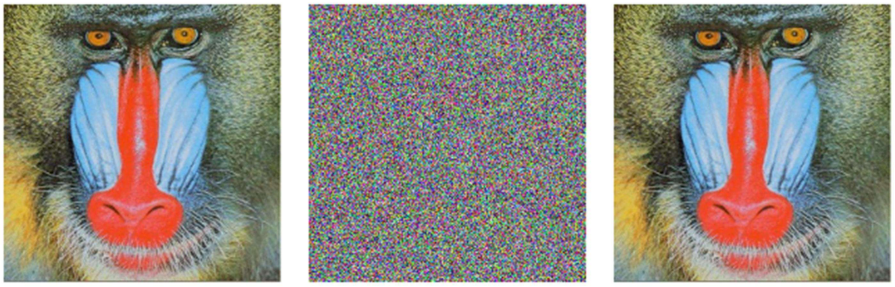
The experimental findings are the only point of discussion when ever the cryptosystem is generated.This chapter is divided in nine sections. Section 6.1 starts with the final results of encryption and decryption algorithms. Section 6.2 outlines the key space and key sensitivity. Section 6.3 covers histogram analysis. Section 6.4 summarises variance analysis. Section 6.5 discusses chi square analysis. Section 6.6 defines entropy analysis. Section 6.7 explains correlation analysis. Section 6.8 clarify UACI and NPCR. Section 6.9 clear up the final PSNR. Section 6.10 is comparative analysis with other operator as well as with standard color image cryptosystems.

6.1 The Final Results

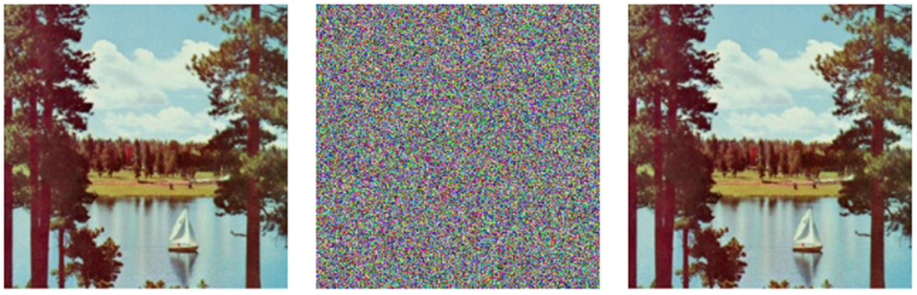
(a)



(b)



(c)



(d)

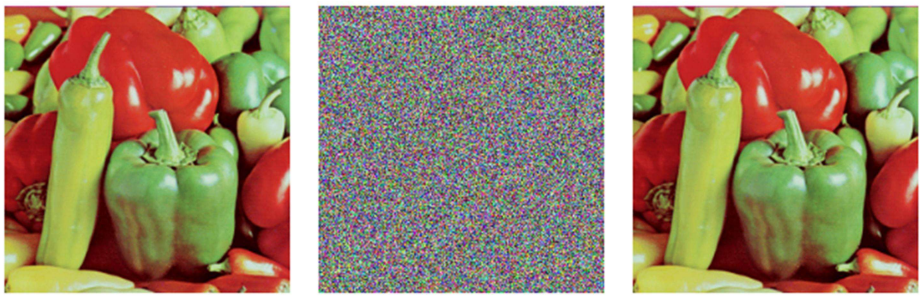


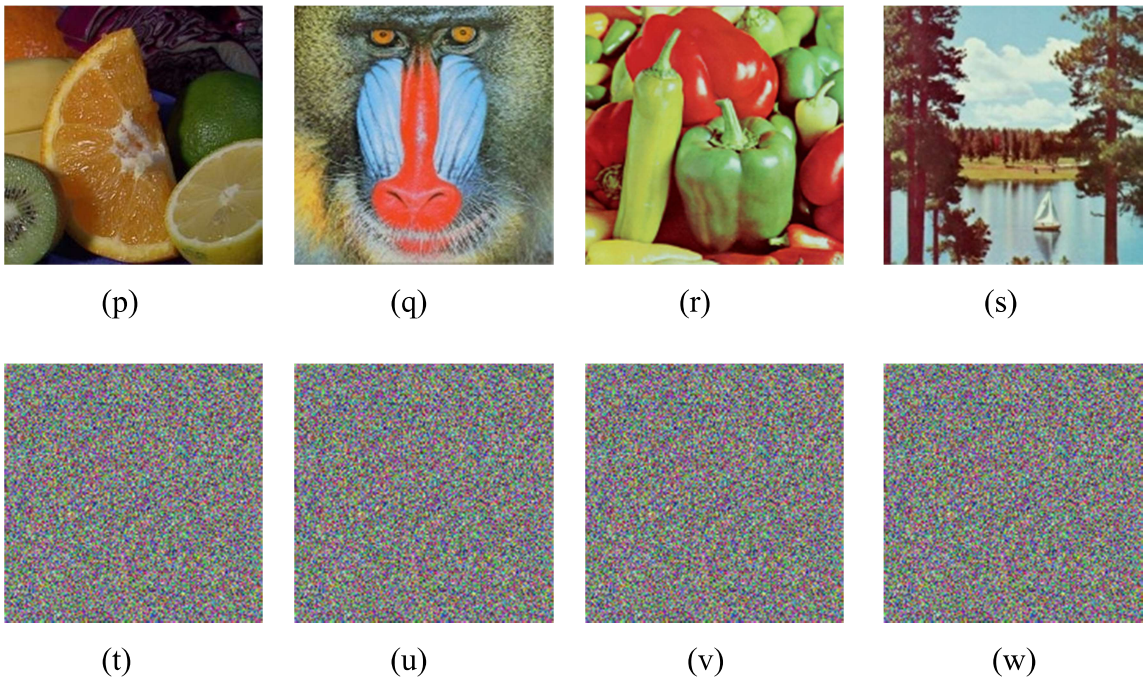
Fig 6.1 Original, en-crypted & de-crypted image of a) Fruit b) baboon c) Pepper d) Lake

6.2 Key Sensitivity and Key length

The robustness of an encryption method against brute-force attacks is often gauged by the length of its key. For high security in cryptographic applications, a key length exceeding 2^{100} bits is generally considered secure. In the cryptosystem we have developed, the key boasts an expansive key space of 2^{240} and has a total length of 240 bits. This symmetric key system utilizes the same key for both encryption and decryption processes.

A critical attribute of any effective encryption system is its high sensitivity to the key. Without this property, even minor deviations in the key used during decryption could fail to produce the correct output image. Ideally, even a slight alteration in the initial key should yield a completely different and unrecognizable decryption output, signifying a robust encryption scheme. In our testing, we demonstrate this sensitivity by modifying a single bit of the key used to encrypt four different RGB color images. We present the decryption results, showcasing the significant discrepancies between the original images (each 256 x 256 pixels) and their decrypted counterparts. This

demonstrates the key's profound impact on the decryption process and underscores the security of our cryptosystem.[3]



(p) (q) (r) (s)

(t) (u) (v) (w)

Fig 6.2. The results of sensitivity of a key of (p)Fruit, (q) Baboon, (r) pepper and (s) Lake. (t) to (w) are decrypted images of respective color images.

6.3 Histogram Analysis

A histogram illustrates the distribution of pixel intensity values across an image. In Fig 6.3, we examine the histograms of both the unencrypted (plain) and encrypted images. The histogram for the encrypted image shows a uniform distribution of pixel values across the entire range from 0 to 255. This uniformity contrasts sharply with the histogram of the plain image, where pixel intensities are distributed unevenly. The single histogram displayed covers all three color channels (Red, Green, and Blue), providing a comprehensive view of the changes in pixel distribution due to the encryption process.

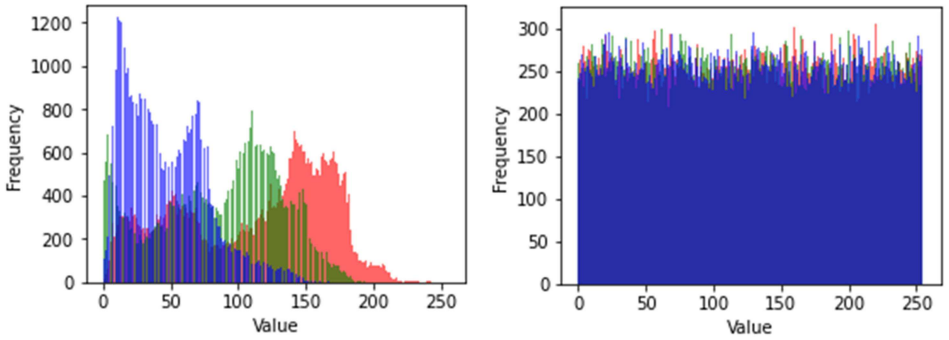
6.4 Variance Analysis

Additionally, we use histogram variances to statistically test the regularity of histograms.

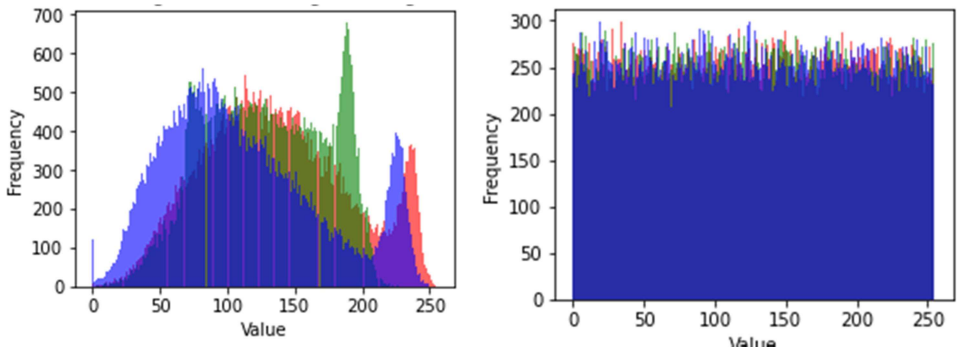
Mathematically, histogram variances are defined as :

$$\text{variance(I)} = \frac{\sum_{x=1}^{m} \sum_{y=1}^{n} (\text{Im(x,y)} - \mu)^2}{m * n}$$

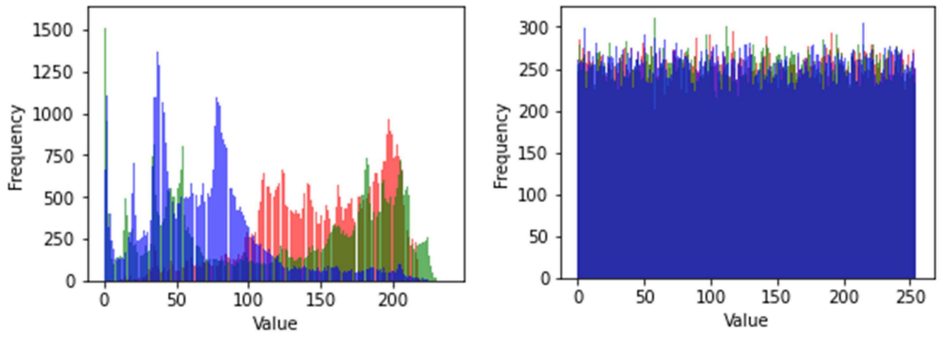
Colour images have low variance since they only have eight bits per pixel and have a non-uniform distribution of pixels in each R, G, and B component image. Table 5 displays natural images with variances less than 3000, while the matching cipher images of the corresponding Re, Gr, and Bl components have variances greater than 5400.



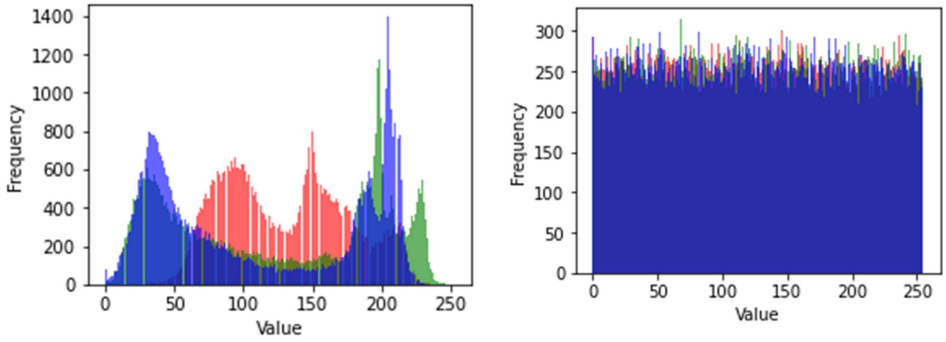
(p)



(q)



(r)



(s)

Fig 6.3. p) plain and cipher histogram of fruit q) plain and cipher histogram of baboon r) plain and cipher histogram of pepper s) plain and cipher histogram of lake.

Table 6.1: Variances of plain and cipher images

Image	Plain Image			Cipher Image		
	Red	Green	Blue	Red	Green	Blue
Fruit	3090.257	3090.257	3090.257	5453.568	5465.366	5475.637
Baboon	2889.991	1972.825	3383.311	5459.752	5444.902	5457.723
Pepper	1995.873	2556.325	1881.152	5469.795	5444.531	5485.011
Lake	1954.858	2572.911	2548.812	5469.953	5498.777	5421.363

6.5 Chi-Square Analysis

The encrypted image's histogram offers a visual depiction of the data, while the chi-Square test offers a statistical analysis of the variation of pixel intensity readings. The chi2 readings for an image of size $m \times n$ and pixel depth p bits can be computed as [15].

$$\chi 2 = \sum_{L=0}^{2^p - 1} \frac{(x^o{}_L - x_e)^2}{x_e}$$

where X_L^o is the frequency that has been observed for the Lth intensity, where L is between 0 $2^p - 1$, and x_e is the frequency expected, which has been computed as:

$$x_e = (2^p - 1)/(m * n)$$

The table ahead displays the values for $\chi 2$ for each of the cipher images. Better pixel intensities are distributed when $\chi 2$ is smaller. From the $\chi 2$-Inverse Cumulative

Distribution Function [12], we can determine the crucial values (upper limit) of $\chi 2$ for 8-bit pictures with level of significances α = 20. These values are 273.79. Since the produced cipher images, $\chi 2$ values are smaller than the crucial values, The hypothesis holds for coming in the range of 20% significance, indicating a uniform variation of pixel magnitude across all of them.

Table 6.2: chi-Square of cipher images

Image	Chi-Square		
	Cipher Image		
	Red	Green	Blue
Fruit	258.06	267.06	268.83
Baboon	232.32	268.43	267.44
Pepper	246.94	216.14	246.22
Lake	257.71	235.82	272.62

6.6 Entropy Analysis

The uncertainty measurement is provided by entropy. The entropy $\varepsilon(S)$ is calculated for a source S that contains n symbols, each with a chance of presence of pi as[16].

$$\varepsilon(S) = \sum_{i=1}^{n} \text{pi} * \log\left(\frac{1}{\text{pi}}\right)$$

Entropy of plain images is low, but that of cipher images should be close to 8 high bits/symbol. Table ahead demonstrates that the crypted images produced by the suggested technique have an entropy larger than 7.9950 , indicating a high degree of randomness. Also table shows entropy of plain images and it can be seen they have uneven entropy.

Table 6.3: Entropy of plain and cipher images

Image	Entropy					
	Plain Image			Cipher Image		
	Red	Green	Blue	Red	Green	Blue
Fruit	7.552	7.337	6.745	7.9971	7.9968	7.9974
Baboon	7.683	7.381	7.682	7.9972	7.9973	7.9971
Pepper	7.271	7.527	7.109	7.9972	7.9975	7.9972
Lake	7.331	7.626	7.341	7.9971	7.9974	7.9973

6.7 Correlation Analysis

Pixels have a strong correlation in plain images, however in crypted images, this correlation should be reduced. One among key statistical features of images is the correlation between neighbouring pixels, which indicates the limit of correlation between pixel values in adjacent positions in the image. The greater the efficiency of the developed encryption system, there is low correlation between the neighbouring pixels of the encrypted picture. Next, 10000 pixels are chosen from both the encrypted and original images in order to examine the pixel correlation. The coefficient of correlation ρ is calculated as [17]

$$\rho(x,y) = \frac{(\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}))}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Consider two vectors of size n, x and y, respectively, with means $\bar{x}$ and $\bar{y}$. $-1 \leq \mu \leq 1$. The original images have a high positive correlation when the values are close to 1, while the cipher images benefit from a low correlation when the values are close to 0.

Table 6.4: Correlation Analysis

Image	Direction	Pixel correlation coefficient					
		Plain Images			Cipher Images		
		Red	Green	Blue	Red	Green	Blue
Fruit	Horizontal	0.9844	0.9879	0.9545	-0.0018	0.0067	-0.0064
	Vertical	0.9856	0.9735	0.9957	-0.0075	0.0018	0.0077
	Diagonal	0.9433	0.9568	0.9402	0.0055	-0.0174	-0.0071
Baboon	Horizontal	0.9126	0.8962	0.9531	0.0031	0.0084	-0.0169
	Vertical	0.6595	0.7867	0.8541	0.0064	-0.0076	-0.0852
	Diagonal	0.7652	0.6806	0.8443	-0.0384	0.0239	0.0259
Pepper	Horizontal	0.9285	0.9465	0.9486	-0.0042	0.0092	-0.0086
	Vertical	0.9725	0.9772	0.9784	-0.0053	0.0041	0.0045
	Diagonal	0.9119	0.9455	0.8801	-0.0345	-0.0127	-0.0057
Lake	Horizontal	0.9322	0.9589	0.9647	0.0042	0.0082	0.0062
	Vertical	0.9794	0.9868	0.9863	-0.0085	-0.0063	0.0058
	Diagonal	0.8935	0.9496	0.9458	-0.002	0.0074	-0.0069

The above table shows correlation analysis for every image in three different direction viz. horizontal i.e. row wise, vertical i.e. column wise and diagonal.

6.8 UACI and NPCR

N: Number of P: Pixel C: Change R: Rate (NPCR) and U: Unified A: Average C: Change I: Intensity (UACI) are important metrics for evaluating the security of a cryptosystem against differential attacks. Consider X and X′ as the cipher images produced by encrypting a plain image of dimensions $m \times n$ with pixel depth p, and then again encrypting after altering a single pixel value in the original image, respectively. The resilience of the encryption method is quantified by calculating NPCR and UACI according to the given formulas.

$$F(u,v) = \begin{cases} 1, & if\ X(u,v) \neq X'(u,v) \\ 0, & if\ X(u,v) = X'(u,v) \end{cases}$$

$$NPCR:\ N(X,X') = \sum_{o,p} \frac{F(u,v)}{m * n} * 100\%$$

$$UACI:\ U(X,X') = \sum_{o,p} \frac{|X(u,v) - X'(u,v)|}{2^p - 1 \cdot (m * n)}$$

where u = 1, 2, …, m; v = 1, 2, …, n & |t| denotes the absolute value of t.

In [13], the optimal settings for UACI and NPCR have been examined. The greatest value of Fmax for a picture with dimensions of m * n and bit depth of p will be $F = 2^p - 1$. The table ahead shows N.P.C.R and U.A.C.I values observed for different images.

Table 6.5: NPCR of cipher images

Images	NPCR		
	NPCR critical value = 99.534077		
	Red	Green	Blue
Fruit	99.6386	99.6118	99.5724
Baboon	99.6062	99.6062	99.6154
Pepper	99.5971	99.5986	99.6121
Lake	99.5994	99.5986	99.6185

From the results of NPCR table 6.5 and UACI table 6.6 we can confirm that our cryptosystem has attained the desired result regarding these two most usefull metrics. These results are best upto the cause.

Table 6.6: UACI of cipher images			
Images	UACI		
	UACI range: 33.1593 - 33.7676, α = 0.005		
	Red	Green	Blue
Fruit	33.3419	33.4274	33.4146
Baboon	33.3902	33.4853	33.2344
Pepper	33.4408	33.4035	33.4181
Lake	33.4693	33.3303	33.4835

6.9 Peak Signal to Noise Ratio – PSNR

Peak Signal to Noise Ratio (P.S.N.R) is a useful metric for evaluating the effectiveness of a cryptosystem against noise attacks and occlusion. Let Pl be the plain image that is encrypted to create the crypted image Z = En(Pl, key) using a key and an encryption technique En. The crypted image is subjected to noise attack or occlusion to produce the image Z′. This image is then decrypted using the matching decoding algorithm Dc and the same key to provide the decrypted image D = Dc(Z′, key). The PSNR is defined in Equation as follows[3,7] if the image has dimensions of m * n and bit depth of p.

$$P.S.N.R = 10 * \log_{10}\left[\frac{(2^p - 1)^2}{\frac{1}{m*n}\sum_{u=1}^{m}\sum_{v=1}^{n}\{D(u,v) - \rho(u,v)\}^2}\right]$$

An image needs a high PSNR—ideally more than 30dB— to be visually discernible. However, because of their random noise like properties and high mean squared error in the associated pixels, cipher images have low PSNR values—typically less than 10dB.

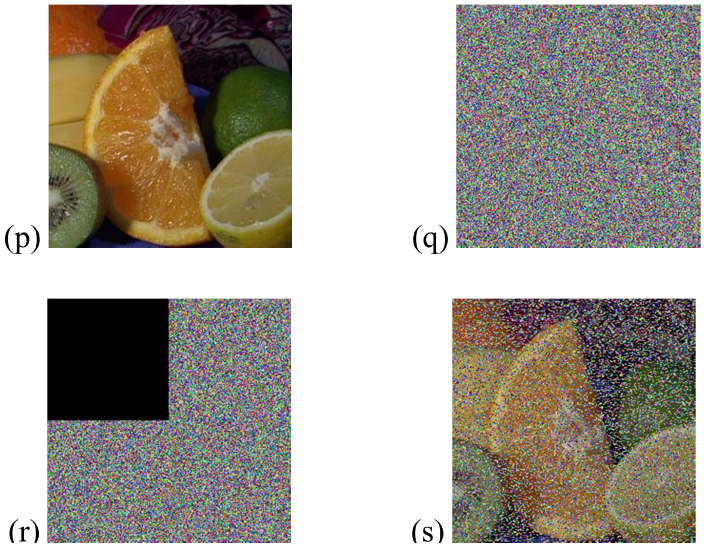


Fig 6.4. Occlusion attack – [a] Original Fruit image, [b] encrypted image, [c] occlusion attack on encrypted image, and [d] decrypted image

Table 6.7: PSNR of cipher images			
Images	PSNR		
	Red	Green	Blue
Fruit	8.23	8.25	8.21
Baboon	8.76	8.76	8.72
Pepper	8.84	8.76	8.76
Lake	8.81	8.81	8.8

6.10 Comparative Analysis

Using the same experimental setup and cryptosystem, the operator ⊗ defined in paper[9] has been compared with other operators such as X⊕R and (+/−) on several color images. The NPCR and UACI tests demonstrate that the operator ⊗ outperforms X⊕R in terms of resistance to differential attacks, since the operator ⊗ yields satisfactory results whereas XOR fails horribly in these tests using with the suggested cryptosystem. While the (+,−) operator yields fairly similar values for UACI and NPCR but also fails in some circumstances as well like for one channel fails and for other two channel give satisfactory result.

However, the main disadvantage of (+/−) operator is removing the pixel correlation and the suggested operator works significantly better in these situations. These findings support the assertion that the operator ⊗ outperforms the standard operators typically employed in traditional image cryptosystems. Table 11 alse provides an overview of various existing cryptosystem's performances and compare our values with theirs values.

Color images can be successfully encrypted using the proposed cryptosystem and the operator ⊗ as we have proposed as well as plain images as shown in paper [9] . The comparison of the findings shows that the proposed cryptosystem performs comparably to the the most modern cryptosystems currently in use.

A thorough comparison of our cryptosystem's performance with that of existing modern cryptosystems reveals that it holds up well. This comparison is based on several key metrics typically used to evaluate the effectiveness of cryptographic solutions, including security against attacks, robustness, and computational efficiency. The results indicate

that our system performs on par with the best available cryptosystems today, offering a reliable option for secure image encryption. This comparative analysis supports the viability of our cryptosystem for practical applications, highlighting its potential to provide robust security solutions for digital imaging needs. Below table shows NPCR UACI comparison.

Table 6.8 Comparative Analysis

Image	Cryptosystem	Encryption Process	NPCR			UACI		
			R	G	B	R	G	B
24 bit color image		Original	~	~	~	~	~	~
24 bit RGB image	proposed	using ⊗	99.638	99.618	99.572	33.342	33.427	33.414
		using XOR	51.416	51.416	51.428	18.392	18.381	18.384
		using + -	93.432	98.182	99.424	33.401	33.421	33.405
24 bit RGB image	Ref [3]	Dynamic DNA encryption and chaos	99.61	99.61	99.61	33.56	33.46	33.45
24 bit color image	Ref [7]	Fractional-Order Hyperchaotic System and DNA Coding	99.61	99.62	99.58	33.47	33.44	33.57
24 bit RGB image	Ref [8]	Fractional Fourier transform and DNA sequence operation	99.56	99.55	99.57	33.41	33.44	33.45
24 bit RGB Image	Ref [4]	hybrid chaotic system and DNA sequences	99.64	99.6	99.61	33.47	33.36	33.44

The values that fail the crucial test are shown by underlining in the NPCR and UACI

Conclusion and Future Scope

In our study, we introduce a cryptosystem specifically designed for encrypting color images using the $\otimes$ operator. The underlying properties of this operator facilitate its dual use in both encryption and decryption processes. This functionality stems from its ability to form an Abelian group, characterized by its distributive and commutative properties. Our cryptosystem incorporates these characteristics in a unique sequence of operations that includes DNA encoding, substitution, and subsequent decoding, intertwined with a specialized mixing procedure preceded by inter channel mixing and mixing rows with initial array mapped from a chaotic map that exploits these mathematical properties of the $\otimes$ operator.

Further, our system is structured to simplify the decryption process by reversing specific changes applied during encryption, thus avoiding the complexity and computational intensity typically associated with multiple rounds of confusion and diffusion found in many current cryptosystems. This streamlined approach relies on basic computational tasks, significantly reducing the operational burden.

The performance of our cryptosystem has been rigorously tested through detailed analysis and experimental trials, confirming that it meets the stringent requirements necessary for a secure image encryption solution. Furthermore, when compared with other commonly utilized operators in the encryption of color images, our results indicate that the $\otimes$ operator provides superior performance, making it an effective tool for robust color image encryption. This comparative analysis underscores the advantages of our approach, highlighting its efficacy and efficiency in securing digital images against potential threats.

Future Scope

- Future work can be done to investigate ways to further strengthen the security of the proposed cryptosystem by incorporating advanced cryptographic techniques or introducing additional layers of encryption to fortify against emerging threats.

- Explore methods to optimize the computational efficiency of the cryptosystem, potentially through parallel processing, algorithmic improvements, or hardware acceleration techniques, to enhance its practical applicability and performance.
- Investigate the integration of machine learning algorithms for improving key management, enhancing security analytics, or optimizing encryption parameters based on dynamic data patterns and user behavior.
- Work towards standardization and interoperability of the cryptosystem to facilitate its adoption across diverse platforms, applications, and industries, ensuring seamless integration and compatibility with existing cryptographic standards and protocols.

REFERENCES

[1] W. Cao, Y. Mao, Y. Zhou, Designing a 2D infinite collapse map for image encryption, Signal Process. (2020) 107457, http://dx.doi.org/10.1016/j.sigpro.2020.107457

[2] Q. Zhang, L. Guo, X. Wei, Image encryption using DNA addition combining with chaotic maps, Math. Comput. Modelling 52 (11–12) (2010) 2028–2035, http://dx.doi.org/10.1016/j.mcm.2010.06.005.

[3] X. Chai, X. Fu, Z. Gan, Y. Lu, Y. Chen, A color image cryptosystem based on dynamic DNA encryption and chaos, Signal Process. 155 (2019) 44–62, https://doi.org/10. 1016/j.sigpro.2018.09.029.

[4] Abolfazl Yaghouti Niyat & Mohammad Hossein Moattar,_Color image encryption based on hybrid chaotic system and DNA sequences https://link.springer.com/article/10.1007/s11042-019-08247-z

[5] X. Chai, J. Bi, Z. Gan, X. Liu, Y. Zhang, Y. Chen, Color image compression and encryption scheme based on compressive sensing and double random encryption strategy, Signal Process. 176 (2020) 107684, http://dx.doi.org/10.1016/j.sigpro.2020.107684.

[6] Z. Liu, C. Wu, J. Wang, Y. Hu, A color image encryption using dynamic DNA and 4-D memristive hyper-chaos, IEEE Access 7 (2019)78367–78378, https://doi.org/10.1109/ACCESS.2019.2922376.

[7] H. Dong, E. Bai, X.-Q. Jiang, Y. Wu, Color image compression-encryption using fractional-order hyperchaotic system and DNA coding, IEEE Access 8 (2020) 163524–163540, https://doi.org/10.1109/ACCESS.2020.30223 98.

[8] M.A.B. Farah, R. Guesmi, A. Kachouri, M. Samet, A novel chaos based optical image encryption using fractional Fourier transform and DNA sequence operation, Opt. Laser Technol. 121 (2020) 105777.

[9] P. Mishra, C. Bhaya, A.K. Pal, A.K. Singh, A novel binary operator for designing medical and natural image cryptosystems, Signal Process. Image Commun. 98 (2021) 116377. https://doi.org/10.1016/j.image.2021.116377

[10] G. Petersen, Arnold cat map survey, Math 45 linear algebra, http://refhub.elsevier.com/S0923-5965(21)00173-9/sb20

[11] Yang, C., Wei, X. and Wang, C., 2021. S-Box design based on 2D multiple collapse chaotic map and their application in image encryption. *Entropy*, *23*(10),p.1312.https://www.mdpi.com/1099-4300/23/10/1312

[12] chi-square inverse cumulative distribution function - MATLAB chi2inv – Math Works India, 2021 ,https://in.mathworks.com/help/stats/chi2inv.html (Accessed 13 May 2021)

[13]] Y. Wu, J.P. Noonan, S. Agaian, et al., NPCR and UACI randomness tests for image encryption, Cyber J.: Multidiscip. J. Sci. Technol. J. Sel. Areas Telecommun. (JSAT) 1 (2) (2011) 31–38. http://refhub.elsevier.com/S0923-5965(21)00173-9/sb26

[14] X. Chai, H. Wu, Z. Gan, D. Han, Y. Zhang, Y. Chen, An efficient approach for encrypting double color images into a visually meaningful cipher image using 2D compressive sensing, Inform. Sci. 556 (2021) 305–340, http://dx.doi.org/10.1016/j.ins.2020.10.007.

[15] X. Wang, L. Feng, H. Zhao, Fast image encryption algorithm based on parallel computing system, Inform. Sci. 486 (2019) 340–358, http://dx.doi.org/10.1016/j.ins.2019.02.049.

[16]] X. Chai, H. Wu, Z. Gan, D. Han, Y. Zhang, Y. Chen, An efficient approach for encrypting double color images into a visually meaningful cipher image using 2D compressive sensing, Inform. Sci. 556 (2021) 305–340, http://dx.doi.org/10.1016/j.ins.2020.10.007.

[17] J. Benesty, J. Chen, Y. Huang, I. Cohen, Pearson correlation coefficient, in: Noise Reduction in Speech Processing, Springer, 2009, pp. 1–4, http://dx.doi.org/10.1007/978-3-642-00296-0_5