CRYPTANALYSIS AND DESIGN OF CHAOS-BASED IMAGE ENCRYPTION SCHEMES

By

WONG KUAN WAI

A thesis submitted to the
Department of Mathematical and Actuarial Sciences,
Lee Kong Chian Faculty of Engineering and Science,
Universiti Tunku Abdul Rahman,
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy (Science)
February 2025

ABSTRACT

CRYPTANALYSIS AND DESIGN OF CHAOS-BASED IMAGE ENCRYPTION SCHEMES

WONG KUAN WAI

Chaos theory has been widely applied in designing image encryption schemes due to its deep connection with cryptographic properties, such as sensitivity to initial conditions and unpredictability. However, many existing image encryption schemes have been shown to be insecure against cryptanalysis. Cryptanalysis is essential for uncovering potential vulnerabilities, as it evaluates the security of encryption schemes, identifies weaknesses, and guides the development of more secure frameworks. Our research reveals that the scheme proposed by Biswas et al. is vulnerable to known plaintext attacks, requiring a time complexity of $2^{264.28}$ encryptions. This is $2^{183.72}$ times faster than brute-force attacks. Similarly, a chosen plaintext attack on Ping et al.'s scheme reveals a reduced key space of $2^{216.51}$, down from the claimed 2^{356}, and identifies inefficiencies in its Henon map-based sequential encryption method. These findings emphasize the need for a more comprehensive analysis of encryption schemes that utilize genetic algorithm and sequential encryption techniques.

To address these challenges, we propose secure image encryption schemes that are based on enhanced chaotic maps. Specifically, we enhance the chaotic behavior of one-dimensional and two-dimensional chaotic maps using cascading techniques. This results in the development of the Logistic-Beta map, the 2D-Henonlog map, and the 2D-Sine-Henon Chaotic Map (2D-SHCM). Furthermore, we present a grayscale image encryption scheme utilizing a permutation-diffusion architecture, as well as a color image encryption scheme based on a genetic algorithm and the 2D-SHCM. Both schemes are designed to ensure high levels of confusion and diffusion in the encrypted images.

Experimental results demonstrate that the proposed schemes effectively resist both statistical and differential attacks. These results highlight the importance of cryptanalysis of existing schemes to identify weaknesses and develop secure encryption methods. The proposed work underscores the need for robust chaotic maps, strong confusion and diffusion mechanisms, and thorough security evaluations as fundamental principles for designing reliable image encryption schemes.

ACKNOWLEDGEMENTS

I would like to express my heartfelt gratitude to all those who have supported me throughout my academic journey.

First and foremost, I offer my heartfelt thanks to my supervisor, Prof. Goi Bok Min, for his invaluable guidance, unwavering support, and constant encouragement. His insightful feedback and belief in my potential have been crucial to the successful completion of this thesis.

I am also profoundly grateful to my co-supervisor, Prof. Yap Wun She, whose expertise and thoughtful advice have greatly enriched my research. I truly appreciate the countless hours he spent discussing ideas, reviewing drafts, and providing constructive feedback. Without his continued support, reaching this milestone would not have been possible.

I am also very thankful to Dr. Denis Wong Chee Keong for his support during my studies. His mentorship during my master's program gave me a strong foundation for my research and sparked my passion for this field. Even after my master's, his encouragement and wisdom have been a huge source of motivation for me.

My sincere appreciation also goes to my co-authors, Prof. Raphael Phan and Prof. Guodong Ye, for their collaboration and contributions to our research papers. Working together has been a rewarding experience, and I appreciate their dedication and insights.

I am deeply grateful to my family for their unconditional love, patience, and constant support throughout this journey. I owe a special thanks to my mother, whose strength and encouragement comforted me during the hardest times. Her belief in me kept me going, and her sacrifices made it possible for me to pursue my dreams. I also wish to express my gratitude to my father, who is now in heaven. He unwaveringly supported our education. His encouragement and love remain a source of inspiration for me every day.

I also thank my brother and sister for their support and belief in me, which gave me strength through tough moments. I am lucky to have them by my side. And to my sister-in-law, my nephew and niece, who brought joy and laughter, lifting my spirits during challenging moments.

I would also like to thank my colleagues and friends, especially Dr. Lee Yap Jia for her emotionally support, Dr. Kuang Kee Seng for his kindly sharing of the thesis template, Dr. Wong Voon Hee for his assistance with the department work, and Dr. Ng Wei Shean, Dr. Chen Huey Voon, Dr. Goh Yong Kheng and Dr. Sim Hong Seng for their valuable feedback in my study. I am also thankful to every member of the “Tuesday Group” and “Temporary Group”. Their support, friendship, and laughter made this journey much more enjoyable. Their encouragement during tough times has meant a lot to me.

Lastly, I extend my appreciation to all those who have contributed to my research in any way. Your assistance and encouragement have helped shape this thesis.

Thank you all.

APPROVAL SHEET

This thesis entitled **"CRYPTANALYSIS AND DESIGN OF CHAOS-BASED IMAGE ENCRYPTION SCHEMES"** was prepared by WONG KUAN WAI and submitted as partial fulfilment of the requirements for the degree of Doctor of Philosophy (Science) at Universiti Tunku Abdul Rahman.

Approved by:

(Ir. Prof. Dr. Goi Bok Min) Date: 4 February 2025
Senior Professor
Department of Mechatronics and BioMedical Engineering
Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

(Prof. Ts. Dr. Yap Wun She) Date: 4 February 2025
Professor
Department of Electrical and Electronic Engineering
Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

LEE KONG CHIAN FACULTY OF ENGINEERING AND SCIENCE

UNIVERSITI TUNKU ABDUL RAHMAN

Date: 4 February 2025

SUBMISSION OF THESIS

It is hereby certified that **WONG KUAN WAI** (ID No: **16UED07691**) has completed this thesis entitled "CRYPTANALYSIS AND DESIGN OF CHAOS-BASED IMAGE ENCRYPTION SCHEMES" under the supervision of Ir. Prof. Dr. Goi Bok Min (Supervisor) from the Department of Mechatronics and BioMedical Engineering, Lee Kong Chian Faculty of Engineering and Science, and Prof. Ts. Dr. Yap Wun She (Co-Supervisor) from the Department of Electrical and Electronic Engineering, Lee Kong Chian Faculty of Engineering and Science.

I understand that the University will upload a softcopy of my thesis in PDF format into UTAR Institutional Repository, which may be accessible to UTAR community and public.

Yours truly,

(WONG KUAN WAI)

DECLARATION

I, WONG KUAN WAI hereby declare that the thesis is based on my original work except for quotations and citation which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UTAR or other institutions.

(WONG KUAN WAI)

Date: 4 February 2025

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Problem Statement

With the advancement of technology, sharing image data on social media platforms such as Instagram, Facebook and WhatsApp has become an essential daily activity in today's society. Image encryption is an important primitive for protecting image data against various types of attacks, preventing unauthorized users from recovering the original image even if the encrypted image is captured. However, the traditional encryption methods such as Data Encryption Standard (DES) (National Bureau of Standards, 1977), Advanced Encryption Standard (AES) (Daemen and Rijmen, 2013) and International Data Encryption Algorithm (IDEA) (Lai and Massey, 1990) are not well-suited for encrypting image data due to the bulky capacity of the images and also the high correlation between the image pixels. These convensional methods are more suitable for textual data rather than image data.

In recent years, chaotic based image encryption becomes a main focal of research in the information and communication security field. Chaotic system has many inherent characteristics such as ergodicity, aperiodicity and highly sensitive to initial conditions and control parameters making it to be popular in designing an image encryption scheme. Therefore, it is widely used in building the permutation matrices, generating a pseudorandom bit sequence which is useful in performing some basic encryption operations, and producing the ciphertext directly when the elements of plaintext are used as the control parameters or initial conditions of the chaotic systems. Many encryption schemes applied the logistic map (May, 1974) due to its simple structure and low computational power requirements. However, its small chaotic range leads to a limited key space, which contributes to weak security. To overcome this drawback, Zahmoul et al. (2017) proposed a one-dimensional chaotic map that based on Beta function which consists of more control parameters. Two-dimensional chaotic maps such as Henon map and modified Henon map proposed by Hua et al. (2020), as well as 2D-CLSS proposed by Teng et al. (2022) based on logistic and sine maps, can widen the chaotic range. However, these chaotic maps still contain periodic windows that could affect their chaotic behavior. Even though the 3D-chaotic map (Bouteghrine et al., 2021) has many control parameters that contribute to a complex chaotic structure, it results in difficult hardware implementation.

The first chaotic based encryption algorithm was proposed by Matthews (1989) and he showed that chaotic system can be applied to cryptography. A secure encryption algorithm must possess the confusion and diffusion functions in its algorithm (Shannon, 1949). Confusion can be attained by obscuring the relationship between cipher-image and the secret key. In other words, every pixel of cipher-image should be affected by secret key as many as possible. Besides, diffusion can reduce the redundancy of the plain-image by spreading it over the cipher-image. It also means that changing a pixel of plain-image will change a large number of pixels of cipher-image. Most of the chaotic based image encryption algorithms are based on permutation-diffusion architecture. It is also known as Fridrich's algorithm because it was firstly proposed by Fridrich (1998). This is the most typical structure that fulfils confusion and diffusion and it had been widely used by other researchers in their ciphers. Pak et al. (2019) proposed a bit-level color image encryption scheme using improved chaotic map on the existing one-dimensional chaotic maps (i.e., logistic map and sine map). The encryption scheme consists of permutation, diffusion, and linear transformation processes. Wu et al. (2018b) applied two-dimensional henon-sine map and DNA coding in designing an image encryption scheme. DNA addition, subtraction and XOR operations are combined to modify the pixel value. However, the schemes proposed in (Pak et al., 2019; Wu et al., 2018b) were vulnerable to the chosen plaintext attack by revealing equivalent encryption elements (Li, Wang, Liu and Fan, 2019; Chen et al., 2020). Ping et al. (2018) proposed an image encryption method based on a two-point diffusion strategy, integrating the permutation and diffusion processes into a single step. The authors claimed that their encryption algorithm is secure against chosen plaintext attacks. However, we discovered the existence of equivalent keys in their scheme, where the same encrypted output can be generated by at least two different keys.

The hybrid model of a chaotic function and genetic algorithm has been widely applied in image encryption algorithms. The crossover and mutation are used as the confusion and diffusion processes, respectively. Pseudorandom bit sequence that generated by chaotic function can be used as the parent bit strings of the crossover process or used in determining the crossover point of the parent bit strings. Wang and Xu (2014) proposed an image encryption scheme based on genetic algorithm and intertwining logistic map. Monte Carlo method is used to choose two parent binary strings. The simple operation of the genetic algorithms could avoid the complexity of using mathematical transformation. Biswas et al. (2015) proposed an image encryption based on N-logistic tent map and genetic algorithms for wireless sensor network. They used mutation and two-point crossover in the encryption algorithms. Later, Das et al. (2018) proposed image encryption scheme based on Arnold cat map and genetic algorithms. Zhang, He, Li and Wang (2020) proposed a color image encryption scheme that combines the two-dimensional non-linear coupling map lattice system with genetic algorithms. Although the genetic algorithm-based image encryption schemes proposed in Biswas et al. (2015); Das et al. (2018); Wang and Xu (2014); Zhang, He, Li and Wang (2020) claimed that their schemes can resist various types of attack and provide sufficient security, we found that the simple operation of genetic algorithm may be vulnerable to known plaintext attack (Biham and Kocher, 1994). Therefore, it is crucial to conduct research on improving the security level of chaotic-based image encryption against various cryptanalytic attacks.

1.2 Objectives

After reviewing recent studies on chaotic systems and image encryption schemes, this research aims to design a secure chaotic based image encryption scheme capable of resisting various cryptanalytic attacks. The three main objectives of this research are as follows.

1. To perform thorough study on how the existing design rules of chaotic based image encryption schemes impact their security. This includes exploring different combinations of permutation and diffusion mechanisms within encryption architectures, as well as cryptanalyzing existing schemes under various cryptanalytic attack models.

2. To propose new chaotic maps using cascading methods to enhance their chaotic behavior. This involves comparing their dynamical performance with the existing chaotic maps and evaluating the effectiveness of the new cascading maps in key generation and encryption processes.

3. To propose improved techniques in constructing a secure chaotic based image encryption scheme. This can be achieved by integrating cascading chaotic maps and cross-plane encryption within a permutation-diffusion architecture.

1.3 Contributions

In this thesis, we conduct cryptanalysis of existing chaotic based image encryption schemes to identify their strengths and vulnerabilities. We propose new chaotic maps that demonstrate better dynamical performance as compared to the existing chaotic maps. Finally, we propose novel image encryption schemes based on the newly proposed chaotic map and the insights gained from the cryptanalysis. The contributions of this thesis are listed as follows:

- **Cryptanalyses of existing chaotic based image encryption schemes.** A known plaintext attack was applied on the image encryption scheme proposed by Biswas et al. (2015). The properties of genetic algorithm were analyzed. The proposed attack methodology can be extended to other encryption schemes designed using genetic algorithm. Besides, a chosen plaintext attack was applied to the image encryption scheme proposed by Ping et al. (2018). The dynamical degradation of the henon map was studied. The security and efficiency of the two-point diffusion strategy were evaluated. Possible enhancements to the attacked ciphers are suggested.

- **Proposals of new chaotic maps.** A cascading technique was applied to develop a new one-dimensional chaotic map derived from the classical logistic map and beta map. Next, a chaotification method was introduced by cascading a two-dimensional chaotic map with a one-dimensional chaotic map using modular operations. The effectiveness of this method was demonstrated using the henon and logistic maps. Besides, the cascading technique was extended to develop a new two-dimensional using sine map and henon map that can address the limitations of discontinuous chaotic ranges encountered in both one-dimensional and high-dimensional chaotic maps. Dynamical performance results indicated the proposed chaotic maps are suitable for pseudorandom number generation and image encryption applications.

- **Proposals of new chaotic based image encryption schemes.** A new grayscale image encryption scheme was developed using a four-dimensional hyperchaotic system and a permutation-diffusion architecture. Lastly, by leveraging the findings from the chaotic maps and cryptanalyses of chaotic based image encryption schemes, this thesis proposes a new color image encryption scheme that fulfills the confusion and diffusion properties necessary for secure image encryption. Experimental results demonstrated that the proposed schemes exhibits strong resistance to the statistical and differential attacks.

1.4 Organization of the thesis

The remainder of this thesis is organised as follows:

- **Literature review.** Chapter 2 provides a comprehensive review of the evolution of image encryption schemes based on the chaotic maps from 1980s to the present. Various chaotification method used in developing chaotic map are discussed. In addition, common cryptanalytic techniques applied to the image encryption schemes are reviewed.

- **Our new cryptanalytic results.** Chapters 3 and 4 present our cryptanalytic results on the image encryption schemes (Biswas et al., 2015; Ping et al., 2018). Each chapter begins with an overview of the respective encryption scheme, then followed by a detailed descruption of our cryptanalytic results. Finally, the chapters conclude with a summary remark.

- **Our new chaotic maps.** Chapters 5 and 6 introduces the newly proposed chaotic maps, including logistic-beta map, 2D-Henonlog map. The mathematical models and dynamical analyses of these chaotic maps are discussed.

- **Our new image encryption schemes.** Chapters 7 and 8 presents the proposed image encryption schemes. Experiment analyses show that the proposed schemes achieve the desired cryptographic properties.

- **Conclusion and future work.** Chapter 9 summarizes results obtained in this thesis and provide suggestions for future research directions.

CHAPTER 2

LITERATURE REVIEW

In recent years, chaos-based image encryption has become a significant area of research within information and communication security. Chaotic systems possess inherent characteristics such as aperiodicity, sensitivity to initial conditions and system parameters, ergodicity, and random-like behavior. These properties make them highly suitable for developing fast and efficient encryption schemes. This chapter begins with an overview of common cryptanalytic methods used to break image encryption schemes. Following that, we present a literature review of various chaos-based image encryption schemes that utilize different architectures.

2.1 Notation

Unless otherwise indicated, most of the notations used in this thesis are listed in Table 2.1.

Table 2.1: Summary of the adopted notations

Notation	**Description**
$\mathbf{A}$	an assembly, which can be a vector, sequence, a matrix and a 2D or 3D image
$\mathbf{A}^{(i)}$	superscript i denotes i^{th} encryption round
$a(i)$ or a_i	i^{th} element of the corresponding 1D assembly $\mathbf{A}$
$a(i,j)$ or $\mathbf{A}_{i,j}$	element or pixel value at i^{th} row and j^{th} column of the corresponding 2D matrix or grayscale image $\mathbf{A}$
$a(i,j,k)$ or $\mathbf{A}_{i,j,k}$	pixel value at i^{th} row and j^{th} column of k^{th} plane of the corresponding 3D color image $\mathbf{A}$
$floor$	rounding a number down to the nearest integer

$\oplus$	bitwise logical exclusively-or (XOR) of two bit strings of the same length

2.2 Image encyption designs

An image encryption is an important primitive that processes a plain image **P** to generate a cipher image **C** using secret key **K**. A secure encryption algorithm must possess the confusion and diffusion functions in its algorithm (Shannon, 1949). Confusion can be attained by obscuring the relationship between cipher-image and the secret key. In other words, every pixel of cipher-image should be affected by secret key as many as possible. Besides, diffusion can reduce the redundancy of the plain-image by spreading it over the cipher-image. It also means that changing a pixel of plain-image will change a large number of pixels of cipher-image. In order to achieve good confusion and diffusion properties, the architecture of the chaotic based image encryption scheme can be designed based on permutation-only, diffusion-only, diffusion-permutation and permutation-diffusion.

In this section, we compare the architectures of various image encryption schemes and their vulnerabilities to different cryptanalytic methods. Before delving into the literature review of various proposals, we introduce common cryptanalytic attack models.

2.2.1 Cryptanalytic attack models

Image encryption is a technique of used to protect visual data by transforming plain images into cipher images. This process is to ensure a secure communication by preventing unauthorized access to the plain images when they are transmitted over public channels. According to Kerckhoffs' Principle, the security of a cryptosystem should rely solely on the secrecy of the key, not on the secrecy of the cryptosystem. This means that while anyone can eavesdrop and obtain the cipher image, the plain image should remain inaccessible without the key. Cryptanalytic attack, or also known as cryptanalysis, refers to the process of recovering the plain image from the cipher image without the key, or even more challenging, deducing the secret key (Petitcolas, 2023).

Attack models specify the information available to an attacker when they attempt to break a cryptographic system. The most common attack models are listed as follows.

1. **Ciphertext-only attack**: The attacker has access to a number of ciphertexts but does not know the corresponding plaintexts. By solely observing these ciphertexts, the attacker attempts to deduce the decryption key or plaintext. Any encryption scheme that is vulnerable to this type of attack is considered to be completely insecure.

2. **Known-plaintext attack**: The attacker has access to a collection of plaintext-ciphertext pairs. The plaintexts are assumed to be randomly selected.

3. **Chosen-plaintext attack**: The attacker has access to the encryption algorithm, allowing them to choose any plaintext and generate the corresponding ciphertext.

4. **Chosen-ciphertext attack**: The attacker has access to the decryption algorithm, even without the secret key, by potentially compromising the decryption equipment, it allows the attacker to select ciphertexts and obtain the corresponding plaintexts. It is important to note that decrypting information is not always sufficient to compromise a system. For example, some video-protection devices allow attackers to perform encryption and decryption queries using the device's chip, but the primary goal of the attacker in such cases is to obtain the key for redistribution. Merely being able to decrypt data without the key may not be enough to break the system.

Next, we will explore image encryption schemes with different designs and their weaknesses against cryptanalytic attacks.

2.2.2 Permutation-only algorithm

Permutation-only image encryption scheme encrypts the images by changing the positions of all the pixels of the image in a secret manner. The permutation process is an invertible function to allow a plain-image to be recovered from the decryption. Let the plain image $\mathbf{P} = \{p(i)\}_{i=1}^{L}$, where $L = M \times N$. Let $\mathbf{W} = \{w(i)\}_{i=1}^{L}$ be the permutation vector with length L. In a permutation-only algorithm, the plain image is encrypted to produce the cipher image $\mathbf{C} = \{c(i)\}_{i=1}^{L}$ by using

$$c(w(i)) = p(i). \tag{2.1}$$

Li et al. (2008) proposed a general quantitative cryptanalysis on the multimedia algorithms against the known- or chosen-plaintexts attacks. The cryptanalysis was achieved by reconstructing the permutation matrix instead of recovering the key. They proved that only $O(\log_L(MN))$ plain-images are needed to break the permutation-only algorithm, where MN is the size of the plain-image in terms of row and column and L is the number of possible different pixel values. The attack complexity of this cryptanalysis is $O(M^2N^2\log_L(MN))$.

Li and Lo (2011) optimized the cryptanalysis in (Li et al., 2008) by adopting a binary tree classification method and a multi-branch tree classification method. With these methods, the permutation-only algorithm can also be broken with $O(\log_L(MN))$ plain-images. However, the spatial and computational complexities are $O(MN)$ and $O(\lceil\log_L(MN)\rceil \cdot MN)$, which are much lower than the attack complexity of the method in (Li et al., 2008). Therefore, the permutation-only algorithm has been proven to be insecure against plaintext attacks based on these cryptanalytic methods.

2.2.3 Diffusion-only algorithm

Diffusion is a substitution function defined by

$$c(i) = p(i) \boxplus f(c(i-1)) \boxplus g(i), \tag{2.2}$$

where $\boxplus$ denotes an arithmetic operation, $\mathbf{G} = \{g(i)\}$ is the diffusion mask made up by chaotic sequences, $f(\cdot)$ is a nonlinear function, and $c(i)$ and $p(i)$ represent cipher pixel and plain pixel, respectively. An image encryption scheme solely based on diffusion operation is considered a less robust design as the confusion property has been neglected.

Ye and Zhou (2014) proposed an image encryption schemes using diffusion-only algorithm. They proposed a block image encryption that depends on double chaotic systems, i.e. Logistic map and 4D hyper-chaotic system. The authors claimed that the diffusion only architecture could overcome the problems inherent in permutation-diffusion process, such as many number of rounds required, permutation process can be easily exploited by known-plaintext attack and chosen-plaintext attack, and the key-dependent problem in the keystream.

However, this architecture was attacked by Yap and Phan (2017) using chosen-plaintext and chosen-ciphertext attacks with the exploitation on the r-round differential with probability of 1. This was also the first attack that demonstrates the vulnerability of image encryption scheme against distinguishing attack. Distinguishing attack is a cryptanalytic method that allows an attacker to distinguish the images encrypted by the underlying encryption algorithm from the random encrypted images. A plaintext-ciphertext pairs with the input differential of $(0, \beta)$ were chosen. If the plaintext-ciphertext were generated by using proposed encryption scheme, then the output difference should also be $(0, \beta)$. The success rate of distinguishing the encrypted images from a truly random images is $1 - 2^{-8p}$, given that size of p pixels is 8-bit long. Besides, Yap and Phan also applied chosen-ciphertext attack on the Ye and Zhou's encryption scheme as the encryption scheme did not satisfy the confusion and diffusion properties due to the linear transformation function of images that uses the modular addition. The authors should investigate how the input difference can influence the output difference under the encryption. To improve the confusion and diffusion properties, adding the addition-rotation-XOR (ARX) operations to the encryption scheme were suggested by Yap and Phan.

Essaid et al. (2019) proposed a novel image encryption algorithm based on a variant of the Hill Cipher and three enhanced one-dimensional chaotic maps, i.e. enhanced logistic map, enhanced chebyshev map and enhanced sine map. The chaotic maps are used to generate the chaotic sequences. The confusion and diffusion process are achieved through the combination of a vector comprising key-pixel pairs and a 2×2 Hill matrix, as well as the addition of a pseudo-random translation vector. However, a comprehensive cryptanalysis conducted by Wen, Lin, Yang and Chen (2024) reveals inherent vulnerabilities in the scheme proposed by Essaid et al., making it susceptible to both chosen-plaintext attack and chosen-ciphertext attack. In a chosen-plaintext attack, the adversary first selects a plaintext image with all pixel values set to zero and obtains the corresponding ciphertext. Through algebraic analysis, they derive an equivalent keystream to compromise the scheme. Next, a plaintext image with all pixel values set to one is chosen, and the resulting ciphertext is used to extract parameters related to the Hill Cipher variant. By combining the findings from these two steps, the original plaintext image can be recovered from any given ciphertext image. Similarly, the scheme is also vulnerable to chosen-ciphertext attacks, which can bypass its security due to these fundamental design flaws. Furthermore, the lack of permutation in the scheme diminishes the algorithm's confusion effect, making it weak and susceptible to attacks.

2.2.4 Diffusion-permutation algorithm

The diffusion-permutation algorithm is constructed using Equations (2.1) and (2.2) as

$$\begin{aligned} b(i) &= p(i) \boxplus f(b(i-1)) \boxplus g(i), \\ c(w(i)) &= b(i), \end{aligned} \tag{2.3}$$

where $\boxplus$ denotes an arithmetic operation, $\mathbf{G} = \{g(i)\}$ is the diffusion mask made up by chaotic sequences, $f(\cdot)$ is a nonlinear function, and $p(i)$, $b(i)$ and $c(i)$

represent plain pixel, diffused plain pixel, and cipher pixel respectively.

According to Wang et al. (2016), diffusion-permutation algorithm is a poorer design as compared to permutation-diffusion due to low key sensitivity. There were two chaotic based image encryption schemes designed based on diffusion-permutation algorithm and were cryptanalyzed by using differential attack.

An image encryption based on a compound chaotic sequence was proposed by Tong and Cui (2008). The compound pseudo-random number sequence generated by two correlated chaotic maps was used to perform XOR substitution of the pixel values. Two chaotic maps were used to perform circular shift position permutations of rows and columns. However, Li et al. (2009) pointed out that there are some defects found in the encryption scheme, making it vulnerable to the differential attack. The weaknesses include insensitivity of the scheme with respect to the changes of plaintexts, existence of weak and equivalent keys, and insufficient randomness of the compound chaotic sequence. Weak keys are referring to some fixed points of the chaotic maps that will affect the randomness of the chaotic sequences, while equivalent keys are referring to some different keys that will result in the same cipher-image, for any given plain-image. Differential chosen-plaintext attack was implemented together with divide-and-conquer (DAC) attack. DAC attack is a method to break the encryption algorithms into two or more smaller components, until these components can be solved easily and directly. In (Tong and Cui, 2008), only three plain-image were required to solve for the row and column circular shift permutations, thereafter the XOR substitution was merely a simple XOR-based stream cipher which can be solved easily.

Dhall et al. (2018) cryptanalyzed a four-round image encryption schemes involving hybrid 1D chaotic systems that made up by linearly combination of logistic map, tent map and sine map (Zhou, Bao and Chen, 2014). Multidimensional chaotic system can improve the security level of the cipher, but the downsides are resulting in the increase of difficulty level of hardware or software implementations and high computation complexity. To overcome this drawback, Zhou, Bao and Chen proposed a new chaotic system that could enhance the chaotic behavior of the chaotic map and also increase the chaotic ranges for the seed maps. Three hybrid chaotic systems suggested by them are Logistic-Tent system, Logistic-Sine system and Tent-Sine system. The four-round encryption scheme involves random pixel insertion, row separation, 1D substitution using Logistic-Tent system, row combination and image rotation. There are many weaknesses found in this encryption scheme by Dhall et al. They performed differential cryptanalysis on four-round encryption scheme without the knowledge of the key. They pointed out that the number of rounds of the encryption scheme was fixed and too small. The permutation step or rotation of the cipher images by 90° counter-clockwise was static and key-independent. There were $4M$ random pixels required to be inserted into M rows of image for each round. Even though the one-time usage of random pixels could provide certain level of security to the cipher, the huge amount of information to be communicated between the sender and receiver was practically infeasible in the real life application. The encryption scheme totally depended on the chaotic behavior of the hybrid chaotic systems and omitted the importance of confusion and diffusion properties in the encryption. To improve this scheme, Dhall et al. suggested to adopt key-based generation of random pixel instead of one-time used pixels. To enhance the confusion properties, key and plaintext-dependent permutation stage is suggested and to be performed before the substitution stage, so that the encryption will follow the permutation-substitution architecture. The fixed and small number of rounds can be solved by introducing a key-dependence of number of rounds with some lower and upper limit. To improve the diffusion properties, instead of having the row-independent substitution process, inter-row feedback can be imposed in 1D-substitution. With these improvements, the desired confusion and diffusion properties of a secure encryption scheme can be satisfied.

In 2016, Xu et al. presented a bit-level image encryption algorithm called BCIEA, based on chaotic maps, which they claimed was secure due to statistical analysis. BCIEA utilizes diffusion-permutation mechanisms, with its security largely depending on the diffusion process that uses cyclic right shifts and bitwise XOR operations. However, Wen, Lin and Feng (2024) later discovered critical security flaws in BCIEA. They found that the chaotic sequences used in BCIEA could act as an equivalent key, weakening its security. Furthermore, the confusion mechanism exhibited regular statistical patterns, making it vulnerable to attacks, particularly an all-zero ciphertext attack. Wen, Lin and Feng also noted that the description of BCIEA was not detailed enough for accurate decryption. As a result, they proposed a chosen-ciphertext attack, which first reduces BCIEA to a diffusion-only algorithm and then uses cipher images with matching sum values to break the confusion mechanism.

2.2.5 Permutation-diffusion algorithm

The permutation-diffusion algorithm is reverse order of diffusion-permutation algorithm. It is represented mathematically by

$$\begin{aligned} d(w(i)) &= p(i), \\ c(i) &= d(i) \boxplus f(c(i-1)) \boxplus g(i), \end{aligned} \tag{2.4}$$

where $\boxplus$ denotes an arithmetic operation, $\mathbf{G} = \{g(i)\}$ is the diffusion mask made up by chaotic sequences, $f(\cdot)$ is a nonlinear function, and $p(i)$, $d(i)$ and $c(i)$ represent plain pixel, permutated plain pixel, and cipher pixel respectively.

Most of the chaotic based image encryption algorithms are based on permutation-diffusion algorithm. It is also known as Fridrich's algorithm because it was firstly proposed by Fridrich (1998). The permutation diffusion operations are presented in the following equation. This is the most typical structure that fulfils confusion and diffusion and it had been widely used by other researchers in their ciphers. However, the permutation function of this kind of the encryption algorithm is independent of plaintext and the diffusion function, therefore it might expose to chosen plaintext attack and chosen ciphertext attack. The one-round encryption scheme based on this design is insecure and can be attacked by differential attacks (Fridrich, 1998; Solak et al., 2010; Fu et al., 2013; Boriga et al., 2014). Fridrich's algorithm with multi-round was attacked by Solak et al. (2010) using the chosen ciphertext attack. However, the attack by Solak et al. is getting harder with the increase of the number of rounds. Some minor defects of the attack proposed by Solak was detected and the attack was further optimized by Xie et al. (2017).

Behnia et al. (2008) proposed a chaotic cryptographic scheme based on two composite polynomial chaotic maps. These two composition maps are used to perform the permutation and substitution processes of the encryption scheme. Li et al. (2010) found that this encryption scheme was vulnerable to the differential attack. The attack involves three steps, breaking confusions I and II, and breaking permutation. The confusions I and II were solved by using the differential cipher-image and also the equivalent key. The remaining permutation process was solved by reconstructing the permutation matrix with $O(\log_L(MN))$ known or chosen plaintexts, where L is the number of different elements in the plaintexts. Some other weaknesses are insufficient randomness of pseudo-randomness number sequences and insensitivity of ciphertext to the change of plaintext.

Zhang et al. (2007) proposed an image encryption scheme using alternate structure (IEAS) based on generalized cat map and one-way coupled map lattice (OCML) in 2007. Zhang et al. (2012) found that the proposed encryption scheme was vulnerable to differential attack. The equivalent secret key could be recovered when the integer parameter is even. Differential cryptanalysis was performed in order to reveal the equivalent secret key of the encryption algorithm by studying the impact of differential plain-image on the differential cipher-image. Some other defects were found in the encryption scheme, i.e., small key space and insensitivity of ciphertext to the change of plaintext due to the implementation of linear operations, such as S-box in the encryption.

Yap et al. (2015) applied impossible differential attack and DAC attack on the image alternate encryption algorithm based on chaotic map which was proposed by Wang and Guo (2014). Yap et al. revisited the key space of Wang and Guo encryption scheme and found that the time complexity for a brute-force attack is $2^{150.053}$ which is smaller than $2^{159.453}$, the key space claimed by Wang and Guo. This shows that the encryption scheme is insecure. Impossible differential attack was applied on 9-round encryption scheme. This cryptanalysis was employing the miss-in-the-middle approach (Biham et al., 1999). Since the number of round, $T = 9$, then there was an 8-round impossible differential with the i-round and j-round differentials with probability of 1, for $i + j = 8$, where the intermediate differences of these two differential were an contradiction. In other words, the probability of i-round differential resulting in j-round differential is zero. Yap et al. also applied a DAC attack on the encryption scheme by using a plain black image. These two methods demonstrated that the image encryption scheme proposed by Wang and Guo was insecure.

Fu et al. (2013) proposed a medical image protection scheme based on chaotic systems. They claimed that bit-level permutation based on discrete cat map has a good confusion properties and able to attain the security level. However, Zhang et al. (2015) later cryptanalyzed the one-round encryption of the proposed scheme. They demonstrate that the bit-level permutation does not practically add the additional strength to the cryptosystem. Zhang et al. also suggested permutation-substitution-permutation architecture could improve the current permutation-substitution structure. The suggestion was later criticized by Chen and Wang (2015) because the permutation-substitution-permutation architecture is insufficient to resist differential attack. Instead of cryptanalyzing on one-round encryption, Chen and Wang performed the differential cryptanalysis on multi-round original scheme and proved that the substitution keystream has no impact on the differential cipher-image and it depends only on the permutation step. They also proposed a new technique called double differential cryptanalysis comparison (DDCC) to attack three or more rounds of encryption.

Boriga et al. (2014) proposed an image encryption scheme based on a two-dimensional hyper-chaotic map that derived from the equations of serpentine curve. The encryption algorithm follows a bi-modular architecture which consists of diffusion and confusion processes and depends on the two serpentine maps. The first serpentine map is adopted to generate random permutation vector and this vector is then used to shuffle the pixels of plain image. The second serpentine map is used to produce two keystreams and the keystreams will be used for the confusion process which alters the pixel values after permutation to reduce the correlation between the plain image and cipher image. A differential attack was performed on this encryption schemes by Wen et al. (2017). selected two special plain-images, P_1 and P_2, in which each pixel of the images was made up by the same value, but the pixel values for P_1 and P_2 are different. This is to eliminate the permutation effect in the algorithm and the encryption became diffusion only algorithm. The diffusion keystreams were revealed by XORing the two cipher-images and the image encryption scheme was broken.

Zhou et al. (2015) proposed an image encryption algorithm based on skew tent map and Line map which adopted a permutation-substitution architecture. The skew tent map was used to generate three chaotic sequences which were used as the secret keys for the permutation and diffusion processes. The binary plain image was permutated using Line map. Chen et al. (2017) applied differential cryptanalysis for one-round encryption with only $M \times N - 1$ chosen plain images. For two-round encryption, Chen et al. applied forward differential and backward differential, or known as two-way differential comparison method in order to obtain permutation matrices for each round. Chen et al. found that the differential cipher-image are independent of the diffusion keys which would substantially reduce the key space of the cryptosystem. The differential cipher-image also depends on a series of linear function of the differential plain-image. If one of the plain images was chosen to be a plain black image with all zero pixels, then the differential cipher-image solely depends on the other plain image and the permutation key. The cryptosystem was broken once the permutation key was revealed. However, permutation matrices for more than 2 rounds are difficult to be obtained by using these two methods. Since the differential cipher-image was formed by linear transformation of the differential plain-image, therefore Chen et al. used codebook attack to break the multi-round encryption algorithm. Codebook attack is a cryptanalytic method that the attacker attempts to construct a "codebook" which is a listing of ciphertexts that correspond to the plaintexts. Chen et al. pointed out three important rules to have a secure permutation-diffusion encryption algorithm, i.e. having a self-synchronous key-stream, permutation process related to plain image, and a nonlinear and complicated diffusion rules.

Hu et al. (2020) proposed a color image encryption algorithm that utilizes a cloud model Fibonacci chaotic system combined with matrix convolution to protect image data. The algorithm began by merging the RGB channels of the original color image and used the generalized Fibonacci sequence to scramble the pixel coordinates. Next, pixel values were substituted through matrix convolution. Finally, forward-backward XOR diffusion was applied between adjacent pixels, and the encrypted image was generated by splitting and reintegrating the three channels.

Liu and Liu (2020) proposed a color image encryption algorithm based on DNA coding and a double chaos system. First, they used the Arnold algorithm to scramble the three color image components, with the number of iterations determined by the average value of these components, enhancing the scrambling effect. Next, they introduced a double chaos system composed of Lorenz chaotic mapping with variable parameters and fourth-order Rossler hyperchaotic mapping to generate three sets of chaotic sequences for diffusion. This double chaos system compensates for the pseudo-randomness of each individual chaotic map, making the sequences more unpredictable. They then transformed both the chaotic component images and chaotic sequences into DNA sequences based on eight DNA coding rules, where the rules are determined by either plaintext information or the generated chaotic sequences. Addition, subtraction, and XOR operations were applied to these DNA sequences. This DNA computation process enables bit-level diffusion for the color images and reduces the overall computational cost.

Recently, Dawahdeh et al. (2018) proposed an encryption scheme that combines elliptic curve cryptography (ECC) and the Hill cipher technique. The scheme's confusion and diffusion architecture is achieved through a 3D Arnold map, ECC, and bit-wise XOR operations. The core concept of this scheme is to transform the Hill cipher from symmetric to asymmetric by using ECC-generated parameters to create the secret key. However, the scheme contains a critical vulnerability related to its secret key, making it susceptible to brute force attacks. With a key space of only 2^{32}, the scheme can be easily compromised using brute force methods, as demonstrated by Lone et al. (2022).

Alexan et al. (2023) proposed a color image encryption algorithm that combines the KAA map with multiple chaotic maps. The algorithm leverages Shannon's principles of security, employing bit-level confusion and diffusion for encryption. Each channel's pixels are shuffled using a sequence generated from the KAA map, while diffusion is achieved through bitwise XOR operations involving two chaotic sequences. The first chaotic sequence is produced by the 2D Sine Logistic Map and the Linear Congruential Generator, while the second is generated using the Bernoulli and Tent chaotic maps. However, it is noted that this simple diffusion process may be vulnerable to plaintext-related attacks.

Zhou and Yu (2024) conducted a comprehensive security analysis of an improved chaos-based image encryption algorithm. The initial algorithm, proposed by Li et al. (2018), involves a permutation process based on the sum of plaintext pixel values and a diffusion process reliant on nine specific pixel values within the permuted image. However, Liu et al. (2019) identified two significant vulnerabilities in the original algorithm: (1) the gray values of the nine specific pixels remain unchanged during the diffusion process, and (2) the permutation process is reversible.

Exploiting these weaknesses, Liu et al. demonstrated that the permuted image could be reconstructed by creating a special plaintext image where the nine specific pixel positions in the permuted image match those of the cipher image. By using the reconstructed permuted image as plaintext, the diffusion process could be attacked to retrieve the permuted image. Since the permutation process is reversible, the original plaintext image could then be recovered entirely. In response, Liu et al. proposed improvements to address these issues, including incorporating a separate permutation step for the nine specific pixels and modifying the original permutation method.

Despite these enhancements, Zhou and Yu revealed that the improved algorithm still contains critical vulnerabilities. Firstly, the improved permutation process introduces equivalent keys, enabling the construction of special plaintexts with identical pixel value sums to those of the original plaintext. This flaw allows the equivalent permutation sequence to be compromised via a chosen-plaintext attack. Secondly, the additional permutation for the specific pixels only permutes these pixels twice in succession, which constitutes a permutation-only encryption. This approach fails to effectively obscure the correlation between adjacent pixels, leaving the encryption scheme insecure.

Recently, Patro et al. (2020) proposed a multiple grayscale image encryption scheme based on cross-coupled chaotic maps, claiming that it could resist known plaintext and chosen-plaintext attacks. The method encrypted multiple images by scrambling them row-wise and column-wise using a permutation table generated by a cross-coupled Piecewise Linear Chaotic Map. Two keys, key_1 and key_2, were used to encrypt the first row or column, followed by a feed-forward XOR operation to generate the cipher image. However, due to its reliance on feed-forward data, all parts of the scrambled image (except the first row and column) could be accurately recovered through a ciphertext-only attack, demonstrating the scheme's insecurity (Singh et al., 2024). Additionally, the high horizontal and vertical correlation inherent in standard images facilitated the effective reversal of the scrambling process. By iteratively matching rows or columns in the scrambled image based on pixel value similarity, the original image structure could be reconstructed without requiring the secret keys. Although the recovered image might not precisely match the original arrangement, the overall information of the multiple images could still be obtained, and reorganization of correlated blocks could produce a near-perfect match. These findings revealed critical vulnerabilities in the Patro et al. scheme, emphasizing the need for more robust encryption designs.

2.3 Summary

From the literature review, we found out that there are some common weaknesses in the chaotic based image encryption schemes and causing the encryption schemes vulnerable to the cryptanalytic attack. The encryption operation involves the following weaknesses should be avoided in the design of a secure chaotic based image encryption scheme. The weaknesses and the suggested improvement are listed as follows.

1. Low sensitivity to the changes of plain-image

   This is the major problem happening in the current image encryption schemes (Ye and Zhou, 2014; Yap et al., 2016; Tong and Cui, 2008; Li et al., 2009; Behnia et al., 2008; Li et al., 2010; Zhang et al., 2007, 2012; Zhou et al., 2015; Chen and Wang, 2015; Patro et al., 2020; Singh et al., 2024). An ideal encryption algorithm should allow a bit of change in the plain-image leading to a large change in the cipher-image. However, linear transformation implemented in the encryption process, such as S-box and XOR operations violate the design rules of nonlinearity of the cryptography. To overcome this problem, nonlinear and complicated operations should be considered in the design of the algorithm (Chen and Wang, 2015).

   A pixel of plaintext can only affect the higher pixel of the corresponding ciphertext and cannot influence other pixels of ciphertexts uniformly. The plaintext-dependent permutation should be implemented in encryption. To link the connection to other row of images, the substitution operation should apply inter-row feedback instead of performing substitution on rows independent of each other (Dhall et al., 2018; Zhou, Bao and Chen, 2014). Besides, problem of independent of keystream from plain-image can be solved by applying the self-synchronous keystream.

2. Existence of equivalent key and weak key

   Equivalent key causes same cipher-image to be generated using a particular plain-image under the encryption of some different keys (Tong and Cui, 2008; Li et al., 2009; Zhang et al., 2007, 2012; Xu et al., 2016; Wen, Lin and Feng, 2024; Essaid et al., 2019; Wen, Lin, Yang and Chen, 2024; Li et al., 2018; Liu et al., 2019; Zhou and Yu, 2024). This could reduce the key space and allow the attacker to access the information of the plain-image easily (Singh et al., 2024). Suppose the differential cipher-image is dependent on a series of functions and the differential plain image. If a special image, $\mathbf{P_1}$ is chosen (e.g. all zero pixels) and with the information of equivalent key, another image $\mathbf{P_2}$ can be recovered by inverting the function. On the other hand, weak key causes the encryption part fails at the certain fixed points of chaotic maps. Therefore, it is important to identify the equivalent and weak keys of the chaotic systems.

3. Differential cipher-image is not related to keystream sequence
   The keystream sequence should not be considered in the cryptanalysis as it could greatly reduce the key space (Fu et al., 2013; Zhang et al., 2015; Chen and Wang, 2015; Zhou et al., 2015; Chen and Wang, 2015). The key-dependent permutation and substitution processes should be implemented.

4. Insufficient randomness of pseudo-random number sequences
   The chaotic system was not a good random number generator based on the random tests (Tong and Cui, 2008; Li et al., 2009; Behnia et al., 2008; Li et al., 2010). Random tests should be performed on chaotic systems to make sure the selected chaotic system can achieve the deterministic pseudo-randomness of the cryptography.

5. Number of rounds of the encryption schemes is fixed and small
   The diffusion and confusion processes could be decrypted easily. Increasing the number of encryption rounds typically strengthens the confusion, diffusion, and avalanche effects in encryption algorithms, thereby enhancing their resilience against cryptographic attacks (Wen, Chen, Yang, Zheng, Wu, Lin, Jian, Lin, Ma, Liu et al., 2024). To overcome this problem, Dhall et al. suggested to implement an alternate forward and backward image encryption algorithms in the substitution stage. Key dependence number of rounds

could also be implemented based on the availability of the resources and security requirements.

In this chapter, we have examined and analyzed existing cryptanalyses related to chaotic-based image encryption. The current security evaluation methods, which primarily rely on quantitative analyses, are inadequate in demonstrating the strength of encryption algorithms against various cryptanalytic attacks. Common weaknesses in the chaotic-based image encryption schemes have been identified and discussed. When constructing encryption algorithms, it is crucial to avoid the poorer designs highlighted in this chapter. To enhance the security of chaotic-based image encryption against cryptanalytic attacks, the following steps will be taken in subsequent chapters:

- Investigate further cryptanalytic attacks that may threaten the security of chaotic-based image encryption.

- Design a secure and efficient chaotic-based image encryption scheme that addresses the identified vulnerabilities.

CHAPTER 3

CRYPTANALYSIS OF GENETIC ALGORITHM-BASED ENCRYPTION SCHEME

This chapter focuses on the cryptanalysis of an image encryption scheme developed by Biswas et al. (2015), which is based on genetic algorithms. The main objective is to explore how the design of chaos-based image encryption schemes affects their security. Through a critical analysis of this design, the chapter aims to identify the weaknesses in the existing method and provide guidelines for creating more robust encryption techniques.

Genetic algorithms, which mimic the process of natural selection to optimize solutions, have been applied in image encryption due to their potential to enhance diffusion and confusion properties. However, our study of the Biswas et al.. scheme reveals significant vulnerabilities. Through a known plaintext attack, we demonstrate that the scheme is low sensitivity to changes in the plain image, which violates the essential cryptographic design rule of nonlinearity. Additionally, the scheme's diffusion mechanism is found to be inadequate, rendering it vulnerable to cryptanalysis.

The findings presented in this chapter emphasize the necessity of integrating more efficient diffusion functions to improve the encryption process. By identifying and addressing the existing weaknesses, this chapter establishes a foundation for proposing an enhanced image encryption scheme that utilizes genetic algorithms while adhering to strong cryptographic principles. The insights gained here not only contribute to the research objective of analyzing current encryption designs but also serve as a stepping stone for developing more secure encryption schemes in the following chapters.

3.1 Introduction

One of the popular image encryption methods is based on genetic algorithms which was proposed by Holland (1975, 1992). This is a method that mimics the natural evolution and selection. Genetic algorithms involve three operations: selection, crossover and mutation. Selection is a process of selecting a portion of the existing population in order to reproduce a new generation. Crossover, or also known as recombination is a process of combining the genetic information of two parents to reproduce a new offspring. Mutation is a process involving a sudden change happens at the genomic level. In image encryption, the genetic information are replaced by the pixel levels of an image.

The evolutionary principles of genetic algorithm can also be applied in searching and optimization. Each solution will be assigned a fitness value. It will be done iteratively by applying these operations until the termination criterion is met. The old population will be replaced by the new population with the optimized fitness value. Therefore, genetic algorithms were used as the optimization method to find the best solution of cipher-image (Abdullah et al., 2012; Enayatifar et al., 2013, 2014) with the entropy as the fitness function. For this method, a specified number of cipher-images are generated using chaotic map. Genetic algorithms are used to modify the cipher-images in order to identify the best cipher image with highest entropy and lowest correlation coefficient. However, this method involves operations with high time complexities. To overcome the weakness, Nematzadeh et al. (2018) modified the genetic algorithm by including a experimental stop criterion.

Besides of the application of optimization, hybrid model of a chaotic function and genetic algorithm has been widely applied in image encryption algorithms. The crossover and mutation are used as the confusion and diffusion processes, respectively. Pseudorandom bit sequence that generated by chaotic function can be used as the parent bit strings of the crossover process or used in determining the crossover point of the parent bit strings. Wang and Xu (2014) proposed an image encryption scheme based on genetic algorithm and intertwining logistic map. Monte Carlo method is used to choose two parent binary strings. The simple operation of the genetic algorithms could avoid the complexity of using mathematical transformation. Biswas et al. (2015) proposed an image encryption based on N-logistic tent map and genetic algorithms for wireless sensor network. They used mutation and two-point crossover in the encryption algorithms. Das et al. (2018) proposed image encryption scheme based on Arnold cat map and genetic algorithms.

Even though the genetic algorithm-based image encryption schemes proposed in (Biswas et al., 2015; Das et al., 2018; Wang and Xu, 2014) claimed that their schemes can resist various types of attack and provide sufficient security, we found that the simple operation of genetic algorithm may be vulnerable to known plaintext attack (Biham and Kocher, 1994). To prove the weakness, we demonstrate the cryptanalysis on the Biswas et al. scheme that applied the two-point crossover operator . From the cryptographic perspective, a scheme is claimed to be vulnerable to a cryptanalytic attack if its time complexity is less than $2^{|K|}$ encryptions, where $|K|$ denotes the length of the secret key K in bits (Yap et al., 2016). We dispute the security claims made by Biswas et al. by showing their scheme is not even secure against known plaintext attack.

Organization: The remainder of this chapter is organised as follows. In the next section, we describe the genetic algorithm and prove that the mutation and crossover are one-to-one operations. We then discuss the image encryption scheme proposed by Biswas et al. in Subsection 3.2.6. In Section 3.3, we present recovery attacks on the keystream and the secret keys against Biswas et al. scheme. Section 3.4 concludes the chapter.

3.2 Preliminaries

Genetic algorithm is an evolutionary algorithm involving three operations, i.e. selection, crossover and mutation. In this section, we explain the descriptions for each stage in details.

3.2.1 Selection

Selection is a process of choosing two parents from population for crossover (Sivanandam and Deepa, 2008). Random selection is used in determining the parent bit string. However, in Biswas et al. scheme, the parent bit strings are selected based on the weight of sub-block that is made up by the pseudorandom bit sequence.

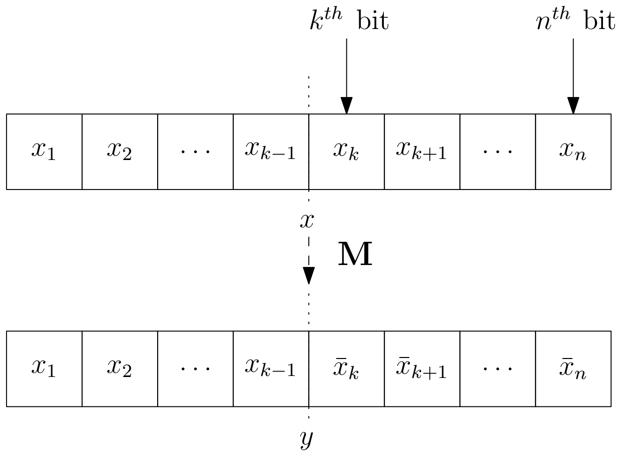
3.2.2 Mutation

Mutation **M** is a process that is normally conducted after crossover (Sivanandam and Deepa, 2008). However, there is an exceptional case, whereby the mutation is used as a substitution function that conducted before the crossover operation. This process is important in disturbing genetic information of the bit strings. **M** is commonly known as a negation operator that changes one or multiple bits in a given bit string.

Let $x = (x_1, x_2, \ldots, x_n) \in \mathbb{F}_2^n$ be a n-bit string. Let k be an arbitrary integer, for $0 \leq k \leq n$. For demonstration, we define the mutation operation as a function that takes two inputs, x and k, and produces an output $y = \mathbf{M}(x,k) = (y_1, y_2, \ldots, y_n) \in \mathbb{F}_2^n$. The mutation operation inverts the bits from the k^{th}- to n^{th}-bit in x, as follows.

- If $k = 0$, then $y = x$;
- Else, $y_i = \begin{cases} x_i, & \text{for } 1 \leq i < k-1, \\ \bar{x}_i, & \text{for } k \leq i \leq n, \end{cases}$

where $\bar{x}_i = 1 - x_i$. Refer to Figure 3.1 for the graphical illustration of **M** operation.



k^{th} bit
n^{th} bit
x_1 x_2 $\cdots$ x_{k-1} x_k x_{k+1} $\cdots$ x_n
x
M
x_1 x_2 $\cdots$ x_{k-1} $\bar{x}_k$ $\bar{x}_{k+1}$ $\cdots$ $\bar{x}_n$
y

Figure 3.1: Graphical illustration of Mutation **M** operation

3.2.3 Crossover

Crossover operation **CO** is a recombining function that creates the child bit strings by exchanging the selected part of their corresponding parent bit stings. There are various types of crossover operation such as single-point, two-point and multi-point crossover. Single-point crossover is a function in which one crossover point is selected and the portion after the crossover points are swapped between two parent bit strings (Kumar and Nirmala, 2012). This method was used in (Das et al., 2018; Hassan and Abuhaiba, 2011; Wang and Xu, 2014).

Apart from single-point crossover, two-point and multi-point crossovers are the generalization of single point crossover, where two or multiple crossover points are selected and the contents between these points are exchanged between two parent bit strings Sivanandam and Deepa (2008). This method was used in the permutation process to change the bit positions (Biswas et al., 2015; Guesmi et al., 2016; Premkumar and Anand, 2018; Ravichandran et al., 2016).

To make our later illustration clearer, we demonstrate one-point crossover operator, in which one crossover point is selected and the contents after the point are exchanged between two parent bit strings. The results obtained can be generalized and applied to the two-point and multi-point crossovers. Let $A = (a_1, a_2, \ldots, a_n)$ and $B = (b_1, b_2, \ldots, b_n)$ be two n-bit strings. Let l be an arbitrary integer, for $0 \leq l \leq n$. We define the crossover operation to be a function that takes A and B to be two parent bit strings and produces two child bit strings $A' = (a'_1, a'_2, \ldots, a'_n)$ and $B' = (b'_1, b'_2, \ldots, b'_n)$ based on the crossover point l, i.e $(A', B') = \mathbf{CO}(A, B, l)$. The crossover operation is described as follows.

- ◦ If $l = 0$, then $(A', B') = (A, B)$.
- ◦ Else,

$$a'_i = \begin{cases} a_i, & \text{for } 1 \leq i < l; \\ b_i, & \text{for } l \leq i \leq n, \end{cases} \quad \text{and}$$

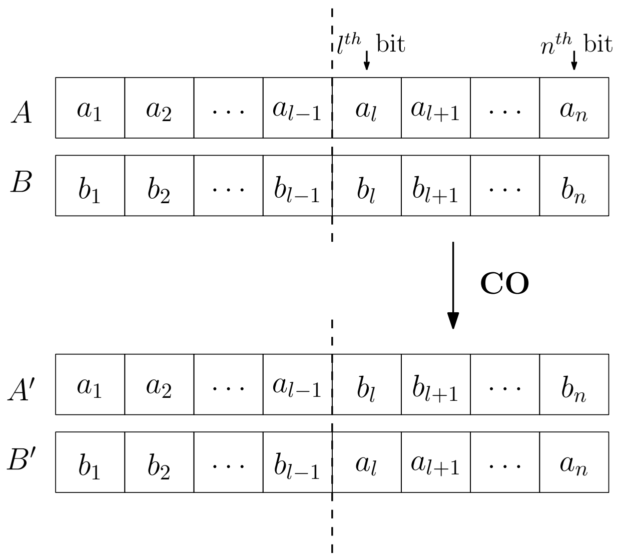
$$b'_i = \begin{cases} b_i, & \text{for } 1 \leq i < l; \\ a_i, & \text{for } l \leq i \leq n. \end{cases}$$

Refer to Figure 3.2 for graphical illustration of single-point Crossover **CO** operation.

The next two propositions show that the genetic algorithm deterministic algorithms.

Proposition 3.2.1: The Mutation **M** is a one-to-one operation.

Proof Recall **M** that described in Subsection 3.2.2. Let



l^{th} bit
n^{th} bit
A
a_1 a_2 $\cdots$ a_{l-1} a_l a_{l+1} $\cdots$ a_n
B
b_1 b_2 $\cdots$ b_{l-1} b_l b_{l+1} $\cdots$ b_n
CO
A'
a_1 a_2 $\cdots$ a_{l-1} b_l b_{l+1} $\cdots$ b_n
B'
b_1 b_2 $\cdots$ b_{l-1} a_l a_{l+1} $\cdots$ a_n

Figure 3.2: Graphical illustration of single-point crossover **CO** operation

$v_1 = (v_{1(1)}, v_{1(2)}, \ldots, v_{1(n)})$ and $v_2 = (v_{2(1)}, v_{2(2)}, \ldots, v_{2(n)})$ be two n-bit strings. For $j = 1,2$, let $v'_j = \mathbf{M}(v_j, k)$ be the output of **M** using the same mutation key k.

To prove **M** is one-to-one, we need to show the following implication is true.

$$\text{If } v'_1 = v'_2, \text{then } v_1 = v_2,$$

where v_1 and v_2 are bit strings of length n. Consider the contrapositive form of this implication:

$$\text{If } v_1 \neq v_2, \text{then } v'_1 \neq v'_2.$$

Our approach is to prove the contrapositive form by using the method of contradiction.

Suppose $v_1 \neq v_2$. Assume $v'_1 = v'_2$. Then, we have

(i) For $1 \leq i < k$, let $v'_{1(i)} = v_{1(i)}$ and $v'_{2(i)} = v_{2(i)}$.

(ii) For $k \leq i \leq n$, let $v'_{1(i)} = \bar{v}_{1(i)}$ and $v'_{2(i)} = \bar{v}_{2(i)}$.

For $1 \leq i < k$, we have $v_{1(i)} = v_{2(i)}$ by comparing $v'_{1(i)}$ and $v'_{2(i)}$ in Part (i). For $k \leq i \leq n$, we flip every $v'_{1(i)}$ and $v'_{2(i)}$ which causing the $v_{1(i)} = v_{2(i)}$ in Part (ii). Therefore, we have $v_1 = v_2$, which is a contradiction. This shows that our assumption is wrong. Thus, $v'_1 \neq v'_2$. Therefore, **M** is a one-to-one operation. $\square$

Proposition 3.2.2: The single-point Crossover **CO** a is one-to-one operation.

Proof Recall **CO** of Subsection 3.2.3. Consider two pairs of parents bit strings (A_1, B_1) and (A_2, B_2) with the same crossover point at l. For $j = 1, 2$, let $(A'_j, B'_j) = \mathbf{CO}(A_j, B_j, l)$ be the output of **CO** using a single crossover point of l.

To prove **CO** is one-to-one, we need to show the following implication is true: If $(A'_1, B'_1) = (A'_2, B'_2)$, then $A_1 = A_2$ and $B_1 = B_2$. Consider the contrapositive form of this implication:

$$\text{If } A_1 \neq A_2 \text{ or } B_1 \neq B_2, \text{ then } (A'_1, B'_1) \neq (A'_2, B'_2).$$

Similar to Proposition 3.2.1, we prove the contrapositive form of this implication by using contradiction.

Suppose $A_1 \neq A_2$ or $B_1 \neq B_2$. Assume $(A'_1, B'_1) = (A'_2, B'_2)$. Let $A_1 = (a_{1(1)}, a_{1(2)}, \ldots, a_{1(n)})$ and $B_1 = (b_{1(1)}, b_{1(2)}, \ldots, b_{1(n)})$. Also, let $A_2 = (a_{2(1)}, a_{2(2)}, \ldots, a_{2(n)})$ and $B_2 = (b_{2(1)}, b_{2(2)}, \ldots, b_{2(n)})$. Then, we have

(i) For $j = 1, 2$ and $1 \leq i < l$, let $a'_{j(i)} = a_{j(i)}$ and $b'_{j(i)} = b_{j(i)}$.

(ii) For $j = 1, 2$ and $l \leq i \leq n$, let $a'_{j(i)} = b_{j(i)}$ and $b'_{j(i)} = a_{j(i)}$.

So, we obtain $a_{1(i)} = a_{2(i)}$ and $b_{1(i)} = b_{2(i)}$, for $l \leq i \leq n$, forcing $A_1 = A_2$ and $B_1 = B_2$ which is a contradiction. Hence, our assumption is wrong. Therefore, $(A'_1, B'_1) \neq (A'_2, B'_2)$. So, **CO** is a one-to-one operation. □

Corollary 3.2.1: Let $\mathscr{M} = \{\mathbf{M}(x, k) \mid x \in \mathbb{F}_2^n, 0 \leq k \leq n\}$ be a set that consists of the output of **M** with the inputs of n-bit string x and mutation key k. Then, $|\mathscr{M}| = n + 1$.

Corollary 3.2.2: Let $\mathscr{L} = \{\mathbf{CO}(A, B, l) \mid A, B \in \mathbb{F}_2^n, 0 \leq l \leq n\}$ be a set that consists of the output of **CO** of two parent bit strings A and B with the single crossover point l. Then, $|\mathscr{L}| = n + 1$.

To show the weaknesses of genetic algorithms, we applies the cryptanalysis on Biswas et al. image encryption scheme in the next section.

3.2.4 Key establishment phase

In this phase, a large key pool is generated by using the elliptic curve over prime field p which is defined by

$$y^2 \bmod p = x^3 + \alpha x + \beta \bmod p, \tag{3.1}$$

where α and β are the coefficients and $x, y \in \mathbb{F}_p$. A number of secret keys where each secret key, denoted as $k_i = (x_{i,0}, y_{i,0})$ for $i > 0$, is shared between two sensor nodes. Each key is referred to as an elliptic curve point which is generated by using Equation (3.1). All of these elliptic curve points form a key pool. When a node wishes to transmit data to another node, it randomly selects a point from its key pool and generates a hash digest of such a point. This hash digest will then be transmitted to the destination node. Upon receiving the hash code, the destination node can retrieve the selected point by matching the received hash digest with the hash digest generated for each point of its shared key pool.

3.2.5 Generation of pseudorandom bit sequence

An N-logistic tent map (Fang et al., 2008), a chaotic map which deals with *integer* parameters, is used to generate pseudorandom bit sequences. The control parameters (i.e., μ, β, m and N) are pre-distributed securely among all sensor nodes in the wireless sensor network whereas the initial conditions (i.e., $x_{i,0}, y_{i,0}$) are the elliptic curve points selected during the key establishment phase as explained in subsection 3.2.4. More precisely, the pseudorandom bit sequences (i.e., x sequence and y sequence) are generated by

$$x_{i,d+1} = \mu x_{i,d}\big(N - \frac{x_{i,d}}{m}\big)/N - \frac{y_{i,d+1}}{2}, \tag{3.2}$$

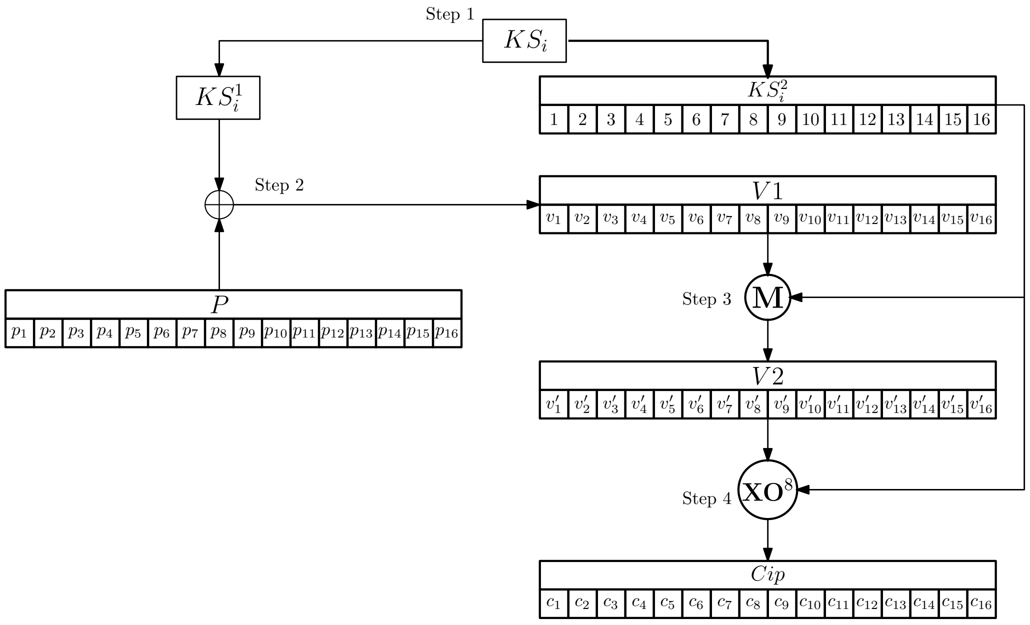
$$y_{i,d+1} = \beta(N - |N - y_{i,d}|), \tag{3.3}$$

where $x = \{x_{i,j}\}_{j=0}^{\infty} \in (0, m \times N)$, $y = \{y_{i,j}\}_{j=0}^{\infty} \in (0, 2 \times N)$, $\mu \in [0,4]$, $\beta \in [1,2]$, $N = [1, 2^{128}]$, $m \in [1, 2^{64}]$ and d is the number of chaotic map iterations. Note that Biswas et al. claimed that the key space of their proposed scheme is around 2^{448}, where x, y, μ, β, N and m are integers. More precisely, x, y and N are with 128-bit length and m is with 64-bit length. Meanwhile, μ and β can be ignored due to smaller key space after being fixed as integers only. As Biswas et al. treated x and y as integers only, thus we assume that the x and y sequences consist of integers value only.

From now onward, we use **M** to represent mutation operation and **XO** to represent two-point crossover operation. Let the subscript j be an integer modulo of 16, i.e. j mod 16. Since we only consider $1 \leq j \leq 16$, if $j = 0(\bmod 16)$, then without loss of generality, we replace $j = 0$ with $j = 16$.

3.2.6 Encryption process

The encryption process proposed by Biswas et al. (shown in Figure 3.3) consists of three main operations, i.e. exclusively-or **XOR** ($\oplus$), **M** and **XO** (Biswas et al., 2015). In Subsection 3.2.5, pseudorandom bit sequence is generated based on the point chosen randomly in each session. The sequence is divided into 256-bit blocks denoted as KS_i for $i > 0$. Each 256-bit block is needed to encrypt every 128-bit plaintext P to a 128-bit ciphertext Cip. The overall process is described



Step 1
KS_i
KS_i^1
KS_i^2
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Step 2
$V1$
v_1 v_2 v_3 v_4 v_5 v_6 v_7 v_8 v_9 v_{10} v_{11} v_{12} v_{13} v_{14} v_{15} v_{16}
P
p_1 p_2 p_3 p_4 p_5 p_6 p_7 p_8 p_9 p_{10} p_{11} p_{12} p_{13} p_{14} p_{15} p_{16}
Step 3
M
$V2$
v'_1 v'_2 v'_3 v'_4 v'_5 v'_6 v'_7 v'_8 v'_9 v'_{10} v'_{11} v'_{12} v'_{13} v'_{14} v'_{15} v'_{16}
Step 4
$\mathbf{XO}^8$
Cip
c_1 c_2 c_3 c_4 c_5 c_6 c_7 c_8 c_9 c_{10} c_{11} c_{12} c_{13} c_{14} c_{15} c_{16}

Figure 3.3: Encryption process proposed by Biswas et al.

as follows:

1. Divide KS_i into two 128-bit sub-blocks, i.e. $KS_i = KS_i^1||KS_i^2$.

2. Compute $V1 = KS_i^1 \oplus P$.

3. Generate $V2$ using $\mathbf{M}$ as follows:

   (a) Divide $V1$ into 16 bytes, i.e. $V1 = v_1||v_2||\ldots||v_{16}$, where v_j denotes the j^{th}-byte of $V1$, for $1 \leq j \leq 16$.

   (b) Similarly, divide KS_i^2 into 16 bytes, i.e. $KS_i^2 = KS_{i,1}^2||KS_{i,2}^2||\ldots||KS_{i,16}^2$, where $KS_{i,j}^2$ denotes the j^{th}-byte of KS_i^2 for $1 \leq j \leq 16$.

   (c) Compute $v'_j = \mathbf{M}(v_j, \sigma_j)$, where $\sigma_j = wt(KS_{i,j}^2)$ for $1 \leq j \leq 16$.

   (d) Obtain $V2$ by concatenating v'_j for $1 \leq j \leq 16$, i.e. $V2 = v'_1||v'_2||\ldots||v'_{16}$.

4. Generate ciphertext Cip using $\mathbf{XO}$ as follows.

   (a) Let $j = 1$, generate two parent 16-bit strings $V_j = v'_j||\ v'_{j+1}$ and $V_{j+2} = v'_{j+2}||v'_{j+3}$.

   (b) Compute $\tau_j = \sigma_j||\ \sigma_{j+1}$ and $\tau_{j+2} = \sigma_{j+2}||\ \sigma_{j+3}$, where σ_j denotes the weight of $KS_{i,j}^2$.

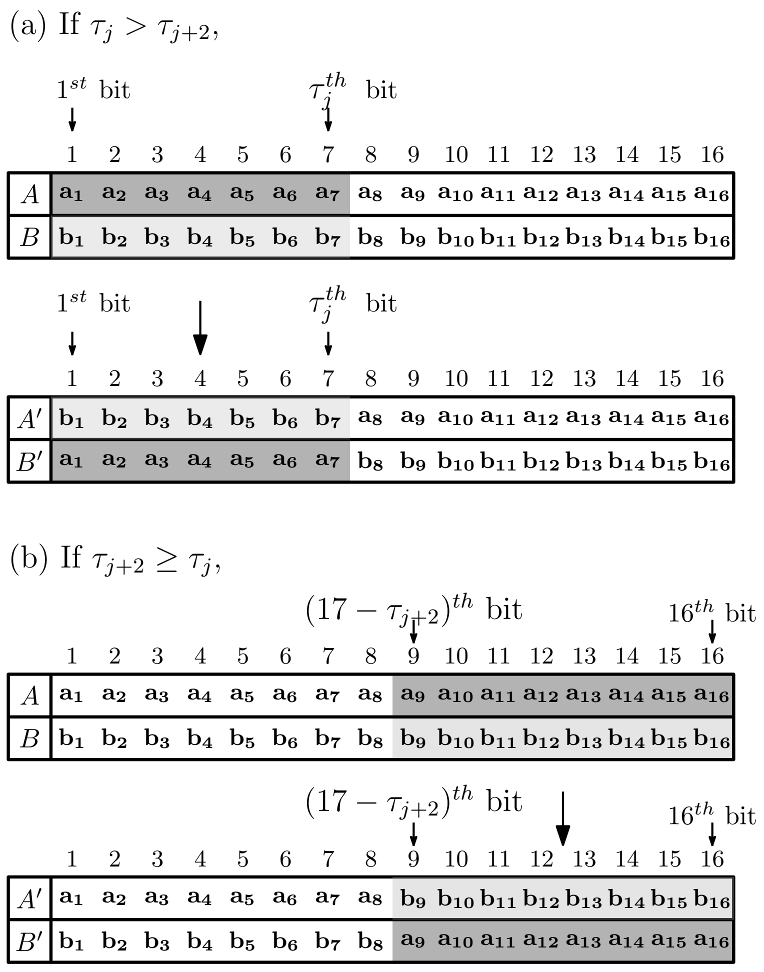

(a) If $\tau_j > \tau_{j+2}$,
1st bit
τ_j^{th} bit
A
B
A′
B′
(b) If $\tau_{j+2} \geq \tau_j$,
$(17 - \tau_{j+2})^{th}$ bit
16th bit

Figure 3.4: Graphical Illustration of Two-Point Crossover Operation

(c) Compute $\mathbf{XO}(V_j, V_{j+2}, \tau_j, \tau_{j+2})$ to generate two child bit strings V'_j and V'_{j+2}. Figure 3.4(a) shows a graphical illustration of $\tau_j = 7$ when $\tau_j > \tau_{j+2}$ while Figure 3.4(b) shows $\tau_{j+2} = 8$ when $\tau_{j+2} \geq \tau_j$.

(d) Obtain $V2$ by concatenating V'_j and V'_{j+2}, i.e. $V2 = V'_j || V'_{j+2}$.

(e) Repeat Step 4(a)-(d) for $j = 3, 5, 7, 9, 11, 13$ and 15. Hence, the **XO** function is repeated for eight times.

5. Obtain $Cip = V2$.

The decryption process is simply the inverse of the encryption process.

3.3 On the security of the encryption process

Instead of recovering the secret key, k_i, we aim to recover the 256-bit block of pseudorandom bit sequence, KS_i, for $i > 0$, provided both plaintext and its corresponding ciphertext are given. In this section, we state some results which will be used in the key recovery attack on Biswas et al. scheme later.

In Biswas et al. scheme, the mutation key k and crossover point l are based on the weight of the second sub-block of KS_i, i.e. KS_i^2. Therefore, Proposition 3.2.1 discussed in Section 3.2 is applied here by replacing k with $wt(KS_{i,j}^2)$ for $i > 0$ and $1 \leq j \leq 16$. While Proposition 3.2.2 is amended as follows.

Proposition 3.3.1: The two-point crossover **XO** is a one-to-one operation.

Proof Recall a two-point crossover can be illustrated as follows: Suppose the two parent bit strings are $A = v'_j||\ v'_{j+1}$ and $B = v'_{j+2}||v'_{j+3}$. Suppose their corresponding bit strings in KS_i^2 are $C = KS_{i,j}^2||KS_{i,j+1}^2$ and $D = KS_{i,j+2}^2||$ $KS_{i,j+3}^2$. The weight of C and D are denoted as τ_C and τ_D, respectively. Then, we compute the child bit strings $(A', B') = \mathbf{XO}(A, B, \tau_C, \tau_D)$.

To prove **XO** is one-to-one, we consider two pairs of parents bit strings (A_1, B_1) and (A_2, B_2), where their corresponding bit strings in KS_i^2 are (C_1, D_1) and (C_2, D_2), respectively. Suppose both $wt(C_1)$ and $wt(C_2)$ equal to τ_C while both $wt(D_1)$ and $wt(D_2)$ equal to τ_D. We need to show the following implication is true: If $(A'_1, B'_1) = (A'_2, B'_2)$, then $A_1 = A_2$ and $B_1 = B_2$. To do so, we consider the contrapositive form of this implication: If $A_1 \neq A_2$ or $B_1 \neq B_2$, then $(A'_1, B'_1) \neq (A'_2, B'_2)$. Similar to Proposition 3.2.2, our approach is to prove the contrapositive form by using contradiction.

Suppose $A_1 \neq A_2$ or $B_1 \neq B_2$. Assume $(A'_1, B'_1) = (A'_2, B'_2)$. For $j = 1, 2$, let $A_j = (a_{j(1)}, a_{j(2)}, \ldots, a_{j(16)})$ and $B_j = (b_{j(1)}, b_{j(2)}, \ldots, b_{j(16)})$. If $\tau_C > \tau_D$, then (A'_1, B'_1) and (A'_2, B'_2) are given by the following conditions. For $j = 1, 2$,

(i) let $a'_{j(i)} = b_{j(i)}$ and $b'_{j(i)} = a_{j(i)}$, for $1 \leq i \leq \tau_C$;

(ii) let $a'_{j(i)} = a_{j(i)}$ and $b'_{j(i)} = b_{j(i)}$, for $\tau_C < i \leq 16$.

Otherwise, if $\tau_D \geq \tau_C$, then (A'_1, B'_1) and (A'_2, B'_2) are given by the following conditions. For $j = 1, 2$,

(iii) let $a'_{j(i)} = a_{j(i)}$ and $b'_{j(i)} = b_{j(i)}$, for $1 \leq i \leq 16 - \tau_D$;

(iv) let $a'_{j(i)} = b_{j(i)}$ and $b'_{j(i)} = a_{j(i)}$, for $16 - \tau_D < i \leq 16$.

Since $wt(C_1) = wt(C_2)$ and $wt(D_1) = wt(D_2)$, therefore the crossover points are the same. So, in Part (i), we obtain $a_{1(i)} = a_{2(i)}$ and $b_{1(i)} = b_{2(i)}$, for $1 \leq i \leq \tau_C$. In Part (iv), we obtain $a_{1(i)} = a_{2(i)}$ and $b_{1(i)} = b_{2(i)}$, for $16 - \tau_D < i \leq 16$. This forces $A_1 = A_2$ and $B_1 = B_2$, which is a contradiction. Hence, our assumption is wrong. Therefore, $(A'_1, B'_1) \neq (A'_2, B'_2)$. So, **XO** is a one-to-one operation. □

Proposition 3.3.2: Suppose $\mathscr{M}$ is defined in Corollary 3.2.1. For $1 \leq j \leq n$, consider $\mathscr{M}$ takes v_j in $V1$ as the inputs and mutation key is based on $\sigma_j = wt(KS^2_{i,j})$. Then, $|\mathscr{M} = \{M(v_j, \sigma_j) \mid v_j \in \mathbb{F}_2^8, 0 \leq \sigma_j \leq 8, 1 \leq j \leq n\}| = 9^n$.

Proposition 3.3.3: For $j = 2l - 1$ and $1 \leq l \leq 8$, let $V_j = v'_j || v'_{j+1}$ and $\tau_j = \sigma_j + \sigma_{j+1}$. Suppose $\mathscr{N}$ is a set that consists of the outputs of **XO** of two parent bit strings (V_j, V_{j+2}) with the crossover points, (τ_j, τ_{j+2}), for $j = 2l - 1$ and $1 \leq l \leq 8$, i.e. $\mathscr{N} = \{\mathbf{XO}(V_j, V_{j+2}, \tau_j, \tau_{j+2}) \mid V_j, V_{j+2} \in \mathbb{F}_2^{16}, 0 \leq \tau_j, \tau_{j+2} \leq 16, j = 2l - 1, 1 \leq l \leq 8\}$. Then, $|\mathscr{N}| = 17^l$.

Proposition 3.3.4: For the worst case scenario, $|wt(KS^2_i)| = 2^{34.14}$.

Proof Recall **XO** that described in Subsection 3.2.6. For $j = 1, 3, 5, \ldots 15$, let $\theta_j = |\tau_j|$ and $\theta_{j+2} = |\tau_{j+2}|$ which can be obtained as follows.

1. If $\tau_j > \tau_{j+2}$, then

    - if $\tau_j \leq 8$, then $\theta_j = \tau_j + 1$ and $\theta_{j+2} = \frac{\theta_j \tau_j}{2}$;

- if $\tau_j > 8$, then $\theta_j = 17 - \tau_j$ and $\theta_{j+2} = 81 - \frac{\theta_j(1+\theta_j)}{2}$.

2. If $\tau_{j+2} \geq \tau_j$, then

   - if $\tau_{j+2} \leq 8$, then $\theta_{j+2} = \tau_{j+2} + 1$ and $\theta_j = \frac{\theta_{j+2}(2+\tau_{j+2})}{2}$;
   - if $\tau_{j+2} > 8$, then $\theta_{j+2} = 17 - \tau_{j+2}$ and $\theta_j = 81 - \frac{\theta_{j+2}(\theta_{j+2}-1)}{2}$.

When $\tau_j > \tau_{j+2}$, θ_j can be calculated by taking all the candidates of σ_j and σ_{j+1} because $\tau_j = \sigma_j + \sigma_{j+1}$. If $\tau_j \leq 8$, then $|(\sigma_j, \sigma_{j+1})| = \tau_j + 1$. Let's say, if $\tau_j = 2$, then $(\sigma_j, \sigma_{j+1}) \in \{(1,1),(2,0),(0,2)\}$. If $\tau_j > 8$, then $|(\sigma_j, \sigma_{j+1})| = 17 - \tau_j$. For instance, if $\tau_j = 9$, then $(\sigma_j, \sigma_{j+1}) \in \{(1,8),(2,7),(3,6),(4,5),(5,4),(6,3),(7,2),(8,1)\}$. On the other hand, to find θ_{j+2}, we sum up all the candidates of (σ_j, σ_{j+1}) from $\tau_j = 0$ until $\tau_j - 1$. Similar explanation applies to the case when $\tau_{j+2} \geq \tau_j$.

The worst case scenario of θ_j and θ_{j+2} is when τ_j and τ_{j+2} are a and 10 for two 16-bit strings in KS_i^2, alternately, where a is a non-negative integer less than 10. There are exactly $(17-10) \times \left(81 - \frac{7(8)}{2}\right) = 7 \times 53 = 371$ values for θ_j and θ_{j+2} in this case. Since the weight of two 16-bit strings in KS_i^2 is known to be 371, therefore the weight of 128-bit KS_i^2 is $371^4 \approx 2^{34.14}$. $\square$

3.3.1 A divide-and-conquer attack

In the traditional security setting of block cipher, same secret key is used in every session to generate same pseudorandom bit sequences. We present a key recovery attack against Biswas et al. scheme under the traditional security setting of block cipher by using divide-and-conquer attack. The attacker is assumed to have the plaintext pairs (P_i, Cip_i), where Cip_i is the corresponding encrypted image of a grayscale image P_i for $i \geq 1$. For ease of understanding, we assume each P_i and each Cip_i are with the length of 128 bits. Based on Propositions

3.3.2-3.3.4, the keystream recovery attack against Biswas et al. scheme can be launched as follows.

1 Consider $KS_1^2 = KS_{1,1}^2||KS_{1,2}^2||\ldots||KS_{1,16}^2$ with weights $\tau_j = \sigma_j + \sigma_{j+1}$, where $\sigma_j = wt(KS_{1,j}^2)$ for $j = 1,3,\ldots,15$. Since the crossover operation is performed using four consecutive bytes, therefore guess the possible values of τ_{13}, τ_{15} and τ_1. Based on Proposition 3.3.3, there are $17^3 = 4913$ possible values for these three weights.

2 For each guess of 4913 possible values of the aforementioned three weights, perform as follows:

   (a) Let $P_1 = p_1||p_2||\ldots||p_{16}$ and $Cip_1 = c_1||c_2||\ldots||c_{16}$.

   (b) Let $A = c_{15}||c_{16}$.

   (c) Let $B = c_{13}||c_{14}$.

   (d) Let $C = c_1||c_2$.

   (e) Let $D = \tau_{15}$.

   (f) Let $E = \tau_{13}$.

   (g) Let $F = \tau_1$.

   (h) Compute $(A',C') = \mathbf{XO}^{-1}(A,C,D,F)$, where $\mathbf{XO}^{-1}$ is the inverse function of $\mathbf{XO}$.

   (i) Compute $(B',A'') = \mathbf{XO}^{-1}(B,A',E,D)$.

   (j) As $A'' = v'_{15}||v'_{16}$, we perform inverse of mutation on v'_{15} and v'_{16} based on the the weight of σ_{15} and σ_{16}. Therefore, guess the possible values of σ_{15} and σ_{16}. Based on the guessed τ_{15}, we compute σ_{15} and σ_{16}. For the worst case scenario, there are $9^2 = 81$ possible values for these two weights based on Proposition 3.3.2. For each guess of the possible values of the aforementioned two weights, compute $KS_{1,j}^1 = \mathbf{M}^{-1}(v'_{kj},\sigma_j) \oplus p_j$, for $j = 15,16$ where $\mathbf{M}^{-1}$ is the inverse function of $\mathbf{M}$.

(k) Repeat Steps 2a to 2j by using other plaintext-ciphertext pairs, i.e. (P_i, Cip_i) for $2 \leq i \leq n$, where the value of n will be discussed in the experiment result in this section later. Store the following data if similar values for $KS^1_{1,15}$ and $KS^1_{1,16}$ for all used plaintext-ciphertext pairs.

- Possible weights of σ_{15} and σ_{16};
- Possible values for $KS^1_{1,15}$ and $KS^1_{1,16}$;
- Store the possible B' and C' and the weights of their corresponding bytes in KS^2_1, i.e. E and F.

(l) Repeat Steps 2j and 2k for the other guesses of possible values of the σ_{15} and σ_{16}.

(m) Repeat Step 2 for the other guesses of 4913 possible values of the aforementioned τ_1, τ_{13} and τ_{15}.

3 Let $j = 13$. Guess the possible values of τ_j. Based on Proposition 3.3.3, there are 17 possible values for this weight.

4 For each guess of 17 possible values of τ_j and also each possible values of (B', E) stored in Step 2, perform as follows:

(a) Let $G = c_{(j-2)} || c_{(j-1)}$.

(b) Let $H = \tau_{j-2}$.

(c) Compute $(G', B'') = \mathbf{XO}^{-1}(G, B', H, E)$.

(d) As $B'' = v'_j || v'_{(j+1)}$, we perform inverse of mutation on v'_j and $v'_{(j+1)}$ based on σ_j and σ_{j+1}. Based on the guessed τ_j, we compute σ_j and σ_{j+1}. For the worst case scenario, there are $9^2 = 81$ possible values for these two weights based on Proposition 3.3.2. For each guess of the possible values of the aforementioned two weights, compute $KS^1_{1,k} = \mathbf{M}^{-1}(v'_k, \sigma_k) \oplus p_k$, for $k = j, j+1$.

(e) Repeat Steps 4a to 4d by using other plaintext-ciphertext pairs, i.e. (P_i, Cip_i) for $2 \leq i \leq n$. Store the following data if similar values for

$KS^1_{1,j}$ and $KS^1_{1,j+1}$ for all used plaintext-ciphertext pairs.

- Possible weights of σ_j and $\sigma_{(j+1)}$;
- Possible values for $KS^1_{1,j}$ and $KS^1_{1,(j+1)}$;
- Store the possible G' and the weight of its corresponding bytes in KS^2_1, i.e. H.

(f) Repeat Steps 4d and 4e for other guesses of the possible values of the σ_j and σ_{j+1}.

(g) Repeat Step 4 for the other guesses of 17 possible values of the aforementioned τ_j and other possible values of (B',E). Store the following data if similar values for $KS^1_{1,j}$ and $KS^1_{1,(j+1)}$ for all used plaintext-ciphertext pairs:

(h) Repeat Step 4 for other guesses of 17 possible values of the aforementioned weight and other possible values of (B',E).

(i) Replace B' and E in Step 4 by G' and H respectively.

5 Repeat Steps 3 and 4 for $j = 11,9,7$ and 5.

6 Consider (C',F) and (B',E) that stored in Steps 2 and Step 5 respectively, where $F = \tau_1$ and $E = \tau_3$. For each possible values of (C',F) and (B',E), perform as follows:

(a) Compute $(C'',B'') = \mathbf{XO}^{-1}(C',B',F,E)$, where $C'' = v'_1||v'_2$ and $B'' = v''_3||v'_4$.

(b) As $C'' = v'_1||v'_2$ $B'' = v'_3||v'_4$, we perform inverse of mutation on v'_j based on σ_j, for $j = 1,2,3,4$. Based on the guessed τ_1 and τ_3 , we compute σ_j for $j = 1,2,3,4$. For the worst case scenario, there are $9^4 = 6561$ possible values for these four weights based on Proposition 3.3.2. For each guess of the possible values of the aforementioned two weights, compute $KS^1_{1,j} = \mathbf{M}^{-1}(v'_j,\sigma_j) \oplus p_j$, for $j = 1,2,3,4$.

(c) Repeat Steps 6a and 6b by using other plaintext-ciphertext pairs, i.e. (P_i, Cip_i) for $2 \leq i \leq n$. Store the following data if similar values of $KS^1_{1,j}$, for $j = 1,2,3$, and 4 for all used plaintext-ciphertext pairs:

- Possible weights of $\sigma_1, \sigma_2, \sigma_3$, and σ_4;
- Possible $KS^1_{1,j}$, for $j = 1,2,3,4$.

(d) Repeat Steps 6b and 6c for other guesses of the possible values of the σ_j for $j = 1,2,3,4$.

(e) Repeat Step 6 for the remaining values of (C', F) and (B', E) stored in Step 2 and Step 5 respectively.

Experiment result: Recovering KS^1_1.

Based on Proposition 3.3.3, $\tau_j = \sigma_j + \sigma_{j+1}$ is determined for $j = 1,3,5,7,9,11,13$ and 15, there are $17^8 \approx 2^{32.70}$ possible values for the weight of $KS^2_{1,j}$ as KS^2_i is made up by eight 16-bit strings. From 17^8 possible values, we deduce the possible σ_j for $1 \leq j \leq 16$ based on Proposition 3.3.4 and there are $7 \times 53 = 371$ possible outcomes for σ_j and σ_{j+1}. Then, each possible values of σ_j will be used to determine $KS^1_{i,j}$ for $1 \leq j \leq 16$ by following Steps 1-6.

We run the computer simulations 50 times by using MATLAB R2017a environment to verify our results. Let the weight of each byte in KS^2_1 be $\sigma_j = wt(u_j) \in \{0,1,2,\ldots,8\}$, for $1 \leq j \leq 16$. Let $\tau = \sigma_j + \sigma_{j+1}$. Let n be the number of plaintext-ciphertext pair. The total number of possible candidates for KS^1_1 is equivalent to the possible values of (σ_j, σ_{j+1}). The worst case scenario for the weights of KS^2_i is $(\sigma_1, \sigma_2, \ldots, \sigma_{16}) = (0,10,0,10,0,10,0,10,0,10,0,10,0,10,0,10)$. We obtained $2^{34.14}$ pairs (KS^1_1, KS^2_1) by using $n = 5$ plaintext-ciphertext pairs which is tallied with the number of possible candidates for KS^2_1 in Proposition 3.3.4.

To calculate the time complexity of $\mathbf{XO}^{-1}$ operation for the experiment, we follow Proposition 3.3.3, i.e. $|\mathcal{N}| = 17^l$, where l is the number of 16-bit of the input bit strings. Steps 2h and 2i use two $\mathbf{XO}^{-1}$ operations to determine τ_1, τ_{13}

and τ_{15}. Therefore, for n plaintext-ciphertext pairs, $n \times 2 \times 17^3$ $\mathbf{XO}^{-1}$ operations are required in these steps. Step 4c is used to find τ_j for $j = 13, 11, 8, 7$, and 5, so it requires $n \times 5 \times 17$ $\mathbf{XO}^{-1}$ operations. Lastly, Step 6a requires n $\mathbf{XO}^{-1}$ operation to identify τ_1 and τ_3. The total time complexity of $\mathbf{XO}^{-1}$ operations is $n(2 \times 17^3 + 5 \times 17 + 1) \approx 2^{13.27}n$ for one encryption.

On the other hand, we follow Proposition 3.3.2 to calculate the time complexity of $\mathbf{M}^{-1}$ operation, i.e. $|\mathscr{M}| = 9^m$, where m is the number of bytes of the input bit strings. For n plaintext-ciphertext pairs, Step 2j requires $n \times 9^2$ $\mathbf{M}^{-1}$ operations to determine σ_{15} and σ_{16}. Meanwhile, Step 4d requires $n \times 5 \times 9^2$ $\mathbf{M}^{-1}$ operations respectively to determine the σ_j, for $j = 13, 11, 8, 7$, and 5. While Step 6b requires $n \times 9^4$ $\mathbf{M}^{-1}$ operations to recover σ_j, for $j = 1, 2, 3, 4$. The total time complexity of $\mathbf{M}^{-1}$ operations is $n(9^2 + 5 \times 9^2 + 9^4) \approx 2^{12.78}n$ for one encryption.

Based on the experiment, we require five plaintext-ciphertext pairs to recover the KS_1^1 on average. By considering $n = 5$, the total time complexity of the proposed attack is $5 \times 2^{13.27} \approx 2^{15.59}$ $\mathbf{XO}^{-1}$ and $5 \times 2^{12.78} \approx 2^{15.10}$ $\mathbf{M}^{-1}$ operations. As one encryption requires 8 $\mathbf{XO}$ and 16 $\mathbf{M}$ operations only, thus the proposed attack has the time complexity around $2^{15.59}/2^3 \approx 2^{12.59}$ encryptions.

3.3.2 Recovering the correct secret key

Recall from Section 3.2.5, we know that pseudorandom bit sequence KS_i is formed by x and y sequences generated by using Equations (3.2) and (3.3). Since $KS_i = KS_i^1 || KS_i^2$, then we let $KS_i^1 = x_{i,d+1}$ and $KS_i^2 = y_{i,d+1}$, where $(x_{i,0}, y_{i,0})$ is the secret key and d is the number of chaotic map iterations to avoid harmful transient effect. We here assume that x and y sequences are integers. Given the parameters $\mu, \beta, N, m, x_{i,t}$, and $y_{i,t}$, one can generate $x_{i,t+j}$

and $y_{i,t+j}$ for any integers i,t, and j.

For ease of understanding, we denote $KS_1^1 = x_{1,1}, KS_1^2 = y_{1,1}, KS_1^3 = x_{1,2}$, and $KS_1^4 = y_{1,2}$, where d is ignored as it is used to avoid harmful transient effect and will not affect the validity of our proposed attack. We perform the following known plaintext-ciphertext attack procedure to recover the remaining secret parameters as follows:

1. From $\mu \in [0,4], \beta \in [1,2], N = [1,2^{128}], m \in [1,2^{64}]$, guess a value for μ, β, N, and m.

2. Based on Proposition 3.3.4, for each guess of (μ, β, N, m) and each pair out of $2^{34.14}$ pairs[1] of (KS_1^1, KS_1^2), compute the values of (KS_1^3, KS_1^4) based on Equations (3.2) and (3.3).

3. Let (P_1, Cip_1) and (P_2, Cip_2) be two different plaintext-ciphertext pairs, where $P_i = p_1||p_2||\ldots||p_{16}$ and $Cip_i = c_1||c_2||\ldots||c_{16}$, for $i = 1$ and 2. Consider $KS_1^2 = KS_{1,1}^2||KS_{1,2}^2||\ldots||KS_{1,16}^2$. Let σ_j be the weight of $KS_{1,j}^2$. Then, perform the following:

   (a) For P_1, compute $v'_j = \mathbf{M}((KS_{1,j}^3 \oplus p_j), \sigma_j)$, for $1 \leq j \leq 16$, where $KS_{1,j}^3$ denotes the j^{th}-byte of KS_1^3.

   (b) For $j = 1$, generate Cip_1 as follows:

      i. Generate two parent 16-bit strings $A = v'_j||v'_{j+1}$ and $B = v'_{j+2}||\ v'_{j+3}$

      ii. Compute two 16-bit strings $\tau_j = wt(KS_{1,j}^4) + wt(KS_{1,j+1}^4)$ and $\tau_{j+2} = wt(KS_{1,j+2}^4 + wt(KS_{1,j+3}^4)$, where $KS_{1,j}^4$ denotes the j^{th}-byte of KS_1^4.

      iii. Compute $(A', B') = \mathbf{XO}(A, B, \tau_j, \tau_{j+2})$ to generate two child bit strings A' and B'.

      iv. Obtain $V2$ by concatenating A' and B', i.e. $V2 = A'||B'$.

      v. Repeat Steps 3b(i) to b(iv) for $j = 3,5,7,9,11,13$, and 15.

      vi. Obtain $Cip'_i = V2$.

[1]We assume worse case scenario to obtain the worse case time complexity of the proposed attack

(c) Repeat Steps 3(a) to (b) for (P_2, Cip_2).

4. If $Cip'_1 = Cip_1$ and $Cip'_2 = Cip_2$, then the guessed values for μ, β, N, m, KS^1_1, and KS^2_1 are correct. If not, then the guessed values are wrong and repeat Steps 1 to 4 until we find the correct guess.

Analysis. For Step 2, the total possible value of (KS^3_1, KS^4_1) is $4 \times 2 \times 2^{128} \times 2^{64} \times (2^{34.14})^2 \approx 2^{263.28}$. For each guess of (KS^3_1, KS^4_1), Step 3a requires 16 **M** operations and Step 3b requires 8 **XO** operations. Overall, the total time complexity of the proposed attack is $2^{263.28} \times 16 \approx 2^{267.28}$ **M** operations and $2^{263.28} \times 8 \approx 2^{266.28}$ **XO** operations. On average, to reduce $2^{263.28}$ possible values of secret key to one, two plaintext-ciphertext pairs (i.e. 256-bit consistency check) are needed. As one encryption requires 8 **XO** and 16 **M**, hence the time complexity needed to recover secret key k_i is around $(2^{267.28} \times 2)/2^4 \approx 2^{264.28}$ encryptions.

Therefore, the time complexity needed to launch such known plaintext attack under traditional security setting of block cipher is around $2^{12.59} + 2^{264.28} \approx 2^{264.28}$ which is much lesser than 2^{448} as claimed by Biswas et al. Therefore, this scheme is vulnerable to the known plaintext attack.

3.3.3 Key recovery attack under defined security setting

In Biswas *et al.* proposed scheme, different secret keys k_i, for $i > 0$, are randomly selected for every session from a same key pool to generate chaotic pseudorandom bit sequences. In each sensor node, a number of secret keys are generated by using elliptic curve operations and then a key pool is formed by these elliptic curve points. If the key pool size is large, then the probability of selecting same secret key is negligible. However, to encrypt an image with the size greater than 128 bits, the pseudorandom bit sequences that generated by the secret key from the same session will continue to be used to encrypt the image. The attack performed in Section 3.3.1 is enough to reveal the secret parameters of the *N*-logistic tent map, i.e. μ, β, N and *m*. The encryption scheme is then broken. Therefore, the cryptosystem is vulnerable to the known plaintext attack by using the keystream generated for one session.

3.4 Summary

This chapter showed that known plaintext attack (which involved divide-and-conquer attack) can be launched against the image encryption scheme designed based on the genetic algorithms. A demonstration of attack is performed on the Biswas et al. and the time complexity required to recover the secret keys is $2^{264.28}$ encryptions. Even though the time complexity is still high, the proposed attack is still faster than brute force attack for $2^{183.72}$ times. The proposed attack showed that the security of an image encryption scheme designed based on genetic algorithms remains unknown and requires further in-depth analysis. The proposed attack and its analysis can be utilized and extended to other image encryption schemes designed based on genetic algorithms.

CHAPTER 4

CRYPTANALYSIS OF AN IMAGE ENCRYPTION SCHEME BASED ON TWO-POINT DIFFUSION STRATEGY AND HENON MAP

This chapter provides a cryptanalysis of an image encryption scheme proposed by Ping et al. (2018), which utilizes a two-point diffusion strategy and the classical Henon map to generate chaotic sequences. The discrete Henon map is employed as the encryption operation. The aim of this chapter is to further investigate the vulnerabilities present in existing encryption methods.

The Ping et al. scheme presents an interesting encryption architecture that integrates permutation and diffusion into a single process, utilizing the Henon map as the primary method for generating the keystream. However, our cryptanalysis, performed through a chosen plaintext attack, reveals significant vulnerabilities in this approach. The scheme's reliance on the Henon map for both permutation and diffusion leads to a sequential encryption process that lacks effective diffusion. This dependency compromises the security of the encryption, as it fails to achieve the necessary level of randomness and resilience against cryptographic attacks.

The findings in this chapter emphasize the need for more robust chaotic maps and better-designed permutation-diffusion strategies to mitigate these vulnerabilities. Building on this analysis, the subsequent chapter will focus on addressing the dynamical degradation of the Henon map and developing improved techniques to enhance its chaotic behavior for more secure encryption methods.

4.1 Introduction

To improve the security of image encryption, proposals utilizing more than one chaotic maps or chaotic maps with higher dimensions were introduced to increase the key space. Two serpentine maps were used by Boriga et al. (2014). One of the serpentine maps was used to generate a random permutation vector which is applied in the permutation process, while another serpentine map was used to generate two keystreams that involve in the diffusion process. Along the same direction, Zhou et al. (2015) applied permutation-diffusion structure in the encryption scheme with the involvement of two chaotic maps: 1) Chaotic sequences generated by skew tent map were used as the secret keys in permutation and diffusion processes and 2) Line map was used to shuffle the pixels of plain-image. Wu et al. (2018b) proposed a new chaotic map, called two-dimensional Henon-Sine map which improves the chaotic behaviours of the underlying chaotic maps. To have high complexity and add more randomness, Julia set fractals and three-dimensional chaotic Lorenz map were applied in the shuffling process of the encryption algorithm (Masood et al., 2020). However, these image encryption schemes are found vulnerable against differential cryptanalysis and chosen plaintext attacks (Wen et al., 2017; Chen et al., 2017, 2020; Munir et al., 2021). Their common mistakes are the differential cipher-images are independent of the diffusion keys but dependent on a series of linear functions of the differential plain-image. So, the permutation effect in the encryption algorithm can be eliminated and the equivalent encryption elements can be found easily.

Even though some chaotic maps demonstrate a good dynamical properties in continuous domain, but the problem of dynamical degradation of chaotic maps in digital domain is inevitable. The performance of chaotic maps jeopardized dramatically when implemented on a limited precision device (for example, digital computer) because the phase space of the chaotic map will be constrained to a finite state phase. It causes the chaotic map to have short cycle length, low complexity and poor randomness. State-mapping network is a graphical method to study the periodicities of a digital map in a quick and accurate way. The dynamical properties of logistic map, tent map, generalized Arnold's Cat map have been studied using the state-mapping network in (Fan and Ding, 2019; Li, Feng, Li, Kurths and Chen, 2019; Li et al., 2021).

Permutation-diffusion architecture is a popular approach used to design a secure image encryption scheme. However, there exist concerns in terms of computational efficiency and security. Many image encryption schemes treat the permutation and diffusion as two stages, therefore the image is processed twice for every round of encryption. Ping et al. (2018) pointed out that the diffusion process is time-consuming and the process cannot be parallelized, therefore it is not applicable in real life. A novel image encryption scheme that is highly optimized for massively parallel architecture was then proposed based on lightweight chaotic maps and simple logical and arithmetic operation by Lee et al. (2018). Ping et al. proposed an image encryption scheme based on two two-dimensional chaotic maps, i.e. a classical Henon map was used to generate a keystream while a discrete Henon map was applied in the encryption algorithm. Instead of having two independent permutation and diffusion processes, they proposed an improved permutation-diffusion process which allows the permutation and diffusion process to intermingle with each other. After calculating the new position of two pixels, the pixel values of these two pixels are changed instead of calculating the position of the next pixel. A two-point diffusion strategy was proposed by Ping et al. to further improve the efficiency of the scheme. This strategy can process two pixels simultaneously, which mean the change of one pixel value will affect its subsequent two pixels. They claimed that this strategy can speed up the spreading process in diffusion and can resist the chosen-plaintext or known plaintext attack because the keystream generation is dependent on the plain images. The key space size is of approximately 2^{356} which can effectively prevent the brute-force attack.

This chapter investigates the two-point diffusion strategy proposed by Ping et al.. Our main contributions can be summarized as follows. Firstly, we show that the scheme is insecure against the chosen plaintext attack even though the key is dependent on the plain image. The equivalent key is revealed by using chosen plaintext attack. Moreover, the attack complexity of the encryption scheme is lower than that of the exhaustive attack. Lastly, the efficiency of two-point strategy has been discussed. We show that the encryption structure is not suitable for parallel computing and suggestion has been given.

Organization: The remainder of this chapter is organised as follows. Section 4.2 describes the encryption algorithms proposed by Ping et al. Section 4.3 demonstrate the detailed cryptanalysis and attacks. In Section 4.4, the efficiency analysis and suggestions are given. The last section concludes the chapter.

4.2 Encryption scheme

The plain image considered in Ping et al.'s scheme is a gray-scale square image with $m = N \times N$ pixels, where N indicates the number of rows or columns of image and m is an even number. Thus, the plain image can be denoted as a square matrix in the domain of $\mathbb{Z}_{256}$, i.e., $\mathbf{P} = [p(i,j)]_{i=0,j=0}^{N-1,N-1}$.

A two-dimensional classical Henon map is used to generate subkeys and a discrete Henon map is used in permutation-diffusion encryption architecture. The definition of these chaotic maps are given as follows.

◇ *Classical Henon map*

$$\begin{cases} x_{d+1} = 1 - ax_d^2 + y_d; \\ y_{d+1} = bx_d, \end{cases} \tag{4.1}$$

where $a = 1.4$ and $b = 0.3$ are the control parameters and d is the d-th iteration of the chaotic map. Lastly, x_0 and y_0 are the initial values of the chaotic map.

◇ *Discrete Henon map*

$$\begin{cases} x_{d+1} = 1 - s_1 x_d^2 + y_d \pmod N; \\ y_{d+1} = x_d + s_2 \pmod N, \end{cases} \tag{4.2}$$

where $s_1, s_2 \in \{1, \ldots, 2^{128}\}$ are the control parameters.

The image encryption algorithm can be divided into two main algorithms, keystream generation and encryption.

4.2.1 Keystream generation

- Secret keys/initial values: $x_0, y_0 \in (0,1)$ with 10^{-15} decimal precision and $s_1, s_2 \in \{1, \ldots, 2^{128}\}$, the control parameters of discrete Henon map given in Equation (4.2).
- Keystream generation consists of the following steps:

1. Compute

$$\text{sum1} = \sum_i \sum_j p(i,j) \pmod{256} \tag{4.3}$$

and

$$\text{sum2} = \sum_i \sum_j p(i,j)^2 \pmod{256}. \tag{4.4}$$

2. Compute the following equations.

$$\begin{aligned}&\text{If } s_1 \leq s_2,\\ &\sigma_1 = \frac{(\text{sum1}+1)\times(\text{sum2}+1)}{257^2} \times \frac{s_1}{s_2}, \quad \sigma_2 = \frac{(\text{sum1}+2)\times(\text{sum2}+2)}{258^2} \times \frac{s_1}{s_2};\\ &\text{If } s_1 > s_2,\\ &\sigma_1 = \frac{(\text{sum1}+1)\times(\text{sum2}+1)}{257^2} \times \frac{s_2}{s_1}, \quad \sigma_2 = \frac{(\text{sum1}+2)\times(\text{sum2}+2)}{258^2} \times \frac{s_2}{s_1}.\end{aligned} \tag{4.5}$$

3. Compute

$$x'_0 = x_0 \pm [\sigma_1 \times 10^5 - \text{floor}(\sigma_1 \times 10^5)] \times 10^{-5}, \tag{4.6}$$

$$y'_0 = y_0 \pm [\sigma_2 \times 10^5 - \text{floor}(\sigma_2 \times 10^5)] \times 10^{-5}. \tag{4.7}$$

   Note that the plus-minus symbol ($\pm$) represents a function that applies addition operation ($+$) if the x'_0 is in the range of (0,1), otherwise subtraction operation ($-$) will be applied.

4. Compute x_{d+k} and y_{d+k}, for $k = 1, \ldots, 3m/2$ by iterating Equation (4.1) with the initial values of (x'_0, y'_0), where d is the number of chaotic map iterations to avoid harmful transient effect.

5. Compute the first keystream $KS^{(1)} = \left\{ks_k^{(1)} \mid ks_k^{(1)} = \text{floor}(|x_{d+k}| \times 10^{15} \pmod{256}), k = 1, \ldots, 3m/2)\right\}$.

6. Compute the second keystream $KS^{(2)} = \left\{ks_k^{(2)} \mid ks_k^{(2)} = \text{floor}(|y_{d+k}| \times 10^{15} \pmod{256}), k = 1, \ldots, 3m/2)\right\}$.

4.2.2 Encryption

Unlike the traditional permutation-diffusion architecture, Ping *et al.* proposed an image encryption allowing the permutation and substitution processed to be done after one another for every two pixels. The author called this method as two-point diffusion strategy. As demonstration, the process flow of the two-point diffusion process for a 2×2 image is shown in Figures 4.1 and 4.2. Process (1a) and (1b) indicated in Figures 4.1 and 4.2 shows the permutation process on how the pixels being changed to a new position. On the other hand, process (2a) and (2b) indicated in Figures 4.1 and 4.2 shows the diffusion process on how the pixel values being changed by the plain pixels and previous cipher values.

As shown in Figure 4.1, the first two pixels $p(0,0)$ and $p(0,1)$ are moved to the location $(0',0')$ and (0”,0”) via process (1a) and (1b). After that, diffusion is done by altering the pixel values and generate two outputs $c^{(r)}(0,0)$ and $c^{(r)}(0,1)$ through process (2a) and (2b). The cipher pixel $c^{(r)}(0,0)$ relies on the permutated pixels $p(0',0')$ and p(0”,0”), whereas $c^{(r)}(0,1)$ relies on $p(0',0')$ only. Referring to process (3) in Figure 4.1, let $c^{(r)}(0,0)$ and $c^{(r)}(0,1)$ be $c(0^*,0^*)$ and $c(0^*,1^*)$, respectively, and they will be used in the encryption for the next two pixels.

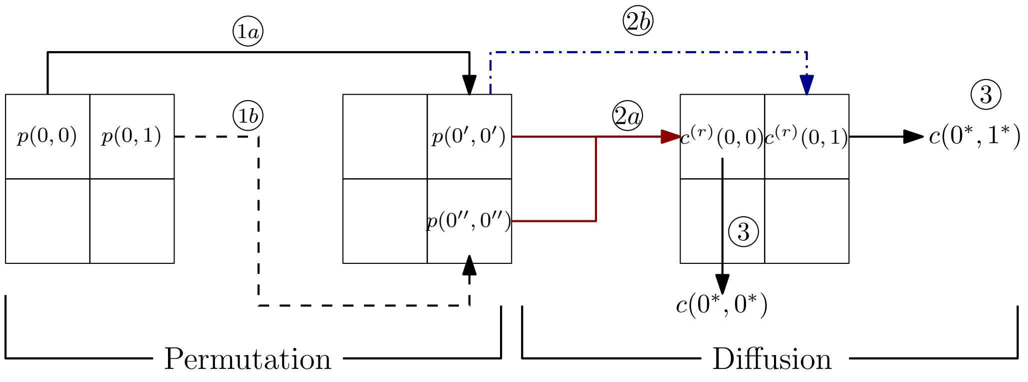


Figure 4.1: One-round encryption for $p(0,0)$ and $p(0,1)$

Same for the next two pixels $p(1,0)$ and $p(1,1)$ in Figure 4.2, they move to new positions $(1',0')$ and (1”,0”) through process (1a) and (1b). As shown in process (2a), the diffusion process to generate $c^{(r)}(1,0)$ involves the permutated pixel $p(1',0')$ and p(1”,0”) that indicated by red solid lines, and the previous cipher pixels $c(0^*,0^*)$ and $c(0^*,1^*)$ that indicated by red dotted lines. For process (2b), $c^{(r)}(1,1)$ relies on $p(1',0')$ that indicated by blue solid line, and the previous cipher pixel $c(0^*,0^*)$ that indicated by blue dotted line. Then, we will let $c^{(r)}(1,0)$ and $c^{(r)}(1,1)$ be $c(0^*,0^*)$ and $c(0^*,1^*)$ which are to be used in the next round.

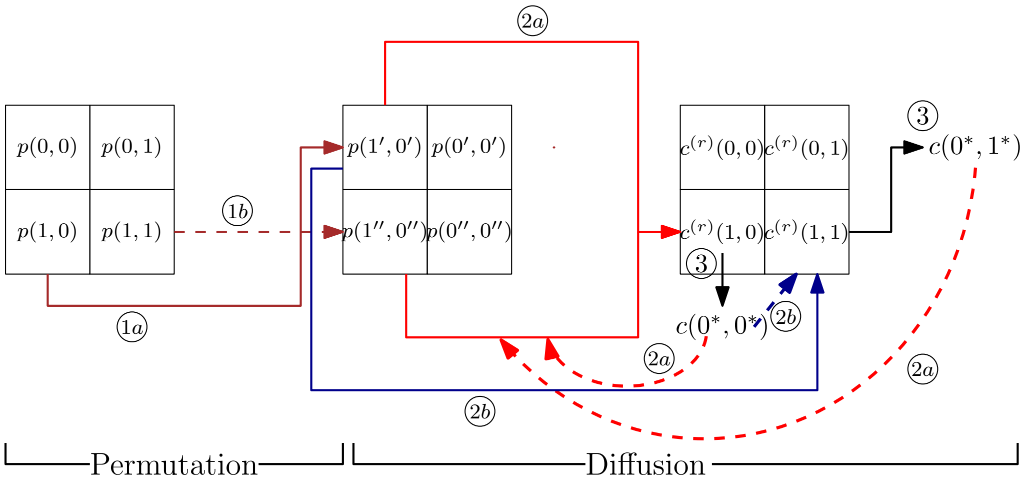


Figure 4.2: One-round encryption for $p(1,0)$ and $p(1,1)$

The overall 3-round image encryption algorithm suggested by Ping *et al.* is illustrated graphically in Figure 4.3. The keystream generation process have been discussed in 4.2.1. The detailed process of the whole image encryption algorithm is given in Algorithm 1.

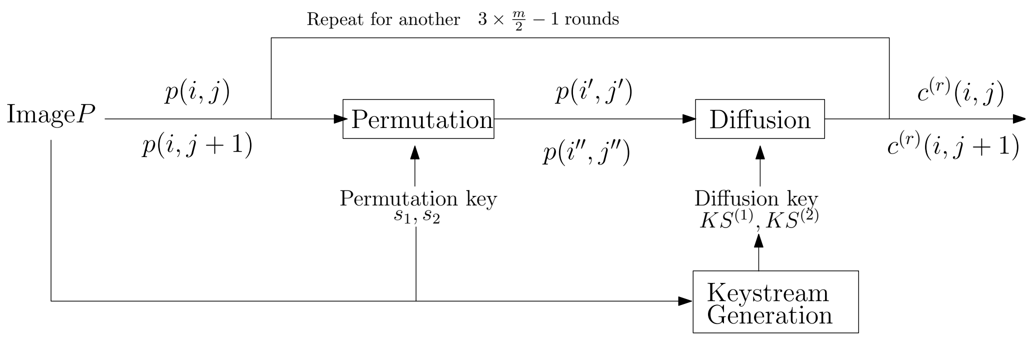


Figure 4.3: The overview of Ping et al. image encryption scheme

4.3 Investigating the security of two-point diffusion strategy

Ping et al. claimed that the encryption algorithm is secure against chosen plaintext attack. However, the authors ignored the existence of equivalent key in

Algorithm 1: Image encryption

```
Input: Plain image P and control parameters of the discrete Henon map $s_1, s_2$
Output: Cipher image $\mathbf{C_m^{(3)}}$
1  Obtain the keystreams $ks_1^{(1)}, \dots, ks_{m/2}^{(1)}$ and $ks_1^{(2)}, \dots, ks_{m/2}^{(2)}$ based on the key stream generation algorithm;
2  $c(0^*, 0^*) = 0$, $c(0^*, 1^*) = 0$, $k = 1$;
3  for $r = 1 : 3$ do
4      for $i = 0 : N - 1$ do
5          for $j = 0 : +2 : N - 1$ do
6              Calculate the new positions $(i', j')$ and $(i'', j'')$ for the two pixels $p(i, j)$ and $p(i, j + 1)$ as follows:
7              $i' = 1 - s_1 i^2 + j \pmod N$;
8              $j' = i + s_2 \pmod N$ ;  Indicated by 1a of Figure 4.1
9              $i'' = 1 - s_1 i^2 + (j + 1) \pmod N$;
10             $j'' = i + s_2 \pmod N$ ;  Indicated by 1b of Figure 4.1
11             Compute cipher pixels $c^{(r)}(i, j)$ and $c^{(r)}(i, j + 1)$ using two-point diffusion operation as follows:
12             $c^{(r)}(i, j) = 1 - ks_k^{(1)}[p(i', j') + c(i^*, j^*)]^2 + p(i'', j'') + c(i^*, j + 1^*) \pmod{256}$;  Indicated by 2a of Figure 4.2
13             $c^{(r)}(i, j + 1) = p(i', j') + c(i^*, j^*) + ks_k^{(2)} \pmod{256}$ ;  Indicated by 2b of Figure 4.2
14             $k = k + 1$;
15         end
16     end
17 end
```

their scheme where the same encryption output can be generated by at least two different keys. From Subsection 4.2.1, we can see that the generation of keystream depends on the sum of image pixels and secret keys. Assuming the secret keys remain the same, there exist more than one plain image producing the same keystream as long as the sum of pixels for two different images are the same.

Besides, the encryption algorithm can be expressed by multiple linear and quadratic equations using modular arithmetic. Therefore, we apply the algebraic attack to recover the keystream and the equivalent key by substituting in the known data for some of the variables (i.e. a number of plaintext-ciphertext pairs). We demonstrate the scheme is insecure by showing the time complexity to break the scheme is less than $2^{|K|}$, where $|K|$ denotes the size of key space (Yap et al., 2016). In this section, we first present the cryptanalysis of the Ping *et al.* scheme by recovering the keystream used in the encryption scheme, with the knowledge of the chosen plain images and their cipher images. After that, we use the recovered keystream to reveal the secret keys $\langle x_0, y_0, s_1, s_2 \rangle$.

4.3.1 Rewriting system of equations

For simplicity, the demonstration of the attack is illustrated based on the smallest square images with the size of 2×2. For $m > 0$, the m^{th} plain image $\mathbf{P_m} = [p_m(i,j)]_{i=0,j=0}^{1,1}$ and the r-round output image $\mathbf{C_m}^{(r)} = [c_m^{(r)}(i,j)]_{i=0,j=0}^{1,1}$ are in the domain of $\mathbb{Z}_{256}$, where $\mathbf{C_m}^{(3)}$ is the cipher image. We first express the encryption algorithm as a system of equations as follows.

First round

$$c_m^{(1)}(0,0) = 1 - ks_1^{(1)}[p_m(1,s_2)]^2 + p_m(0,s_2) \tag{4.8}$$

$$c_m^{(1)}(0,1) = p_m(1,s_2) + ks_1^{(2)}. \tag{4.9}$$

$$c_m^{(1)}(1,0) = 1 - ks_2^{(1)}[p_m(1-s_1,1+s_2) + c_m^{(1)}(0,0)]^2 + p_m(-s_1,1+s_2) + c_m^{(1)}(0,1) \tag{4.10}$$

$$c_m^{(1)}(1,1) = p_m(1-s_1,1+s_2) + c_m^{(1)}(0,0) + ks_2^{(2)}. \tag{4.11}$$

Second round

$$c_m^{(2)}(0,0) = 1 - ks_3^{(1)}[c_m^{(1)}(1,s_2) + c_m^{(1)}(1,0)]^2 + c_m^{(1)}(0,s_2) + c_m^{(1)}(1,1) \tag{4.12}$$

$$c_m^{(2)}(0,1) = c_m^{(1)}(1,s_2) + c_m^{(1)}(1,0) + ks_3^{(2)}. \tag{4.13}$$

$$c_m^{(2)}(1,0) = 1 - ks_4^{(1)}[c_m^{(1)}(1-s_1,1+s_2) + c_m^{(2)}(0,0)]^2 + c_m^{(1)}(-s_1,1+s_2) + c_m^{(2)}(0,1) \tag{4.14}$$

$$c_m^{(2)}(1,1) = c_m^{(1)}(1-s_1,1+s_2) + c_m^{(2)}(0,0) + ks_4^{(2)}. \tag{4.15}$$

Third round

$$c_m^{(3)}(0,0) = 1 - ks_5^{(1)}[c_m^{(2)}(1,s_2) + c_m^{(2)}(1,0)]^2 + c_m^{(2)}(0,s_2) + c_m^{(2)}(1,1) \tag{4.16}$$

$$c_m^{(3)}(0,1) = c_m^{(2)}(1,s_2) + c_m^{(2)}(1,0) + ks_5^{(2)}. \tag{4.17}$$

$$c_m^{(3)}(1,0) = 1 - ks_6^{(1)}[c_m^{(2)}(1-s_1,1+s_2) + c_m^{(3)}(0,0)]^2 + c_m^{(2)}(-s_1,1+s_2) + c_m^{(3)}(0,1) \tag{4.18}$$

$$c_m^{(3)}(1,1) = c_m^{(2)}(1-s_1,1+s_2) + c_m^{(3)}(0,0) + ks_6^{(2)}. \tag{4.19}$$

The cipher pixels at the third round $c_m^{(3)}(0,0)$, $c_m^{(3)}(0,1)$, $c_m^{(3)}(1,0)$ and $c_m^{(3)}(1,1)$ are known under the chosen plaintext attack. All the twelve subkeys ($ks_k^{(1)}$, $ks_k^{(2)}$), for $1 \le k \le 6$ are required to be recovered. In the first round, there are totally 12 unknown variables which are $ks_1^{(1)}$, $ks_1^{(2)}$, $ks_2^{(1)}$ and $ks_2^{(2)}$ and $p_m(0,s_2)$, $p_m(1,s_2)$, $p_m(-s_1,1+s_2)$ and $p_m(1-s_1,1+s_2)$, $c_m^{(1)}(0,0)$, $c_m^{(1)}(0,1)$,

$c_m^{(1)}(1,0)$ and $c_m^{(1)}(1,1)$. The plain and cipher images are unknown because of the unknown of s_1 and s_2.

In the second round, the additional unknown variables are $ks_3^{(1)}$, $ks_3^{(2)}$, $ks_4^{(1)}$ and $ks_4^{(2)}$, $c_m^{(1)}(0,s_2)$, $c_m^{(1)}(1,s_2)$, $c_m^{(1)}(-s_1,1+s_2)$, $c_m^{(1)}(1-s_1,1+s_2)$, $c_m^{(2)}(0,0)$, $c_m^{(2)}(0,1)$, $c_m^{(2)}(1,0)$ and $c_m^{(2)}(1,1)$. While in the third round, the additional unknown variables are $ks_5^{(1)}$, $ks_5^{(2)}$, $ks_6^{(1)}$ and $ks_6^{(2)}$, $c_m^{(2)}(0,s_2)$, $c_m^{(2)}(1,s_2)$, $c_m^{(2)}(-s_1,1+s_2)$, and $c_m^{(2)}(1-s_1,1+s_2)$.

Hence, there are 32 unknown variables and 4 known variables ($c_m^{(3)}(0,0)$, $c_m^{(3)}(0,1)$, $c_m^{(3)}(1,0)$ and $c_m^{(3)}(1,1)$) in total. To solve these unknown variables, there must be at least as many equations as the number of variables. With four images, we can identify the unknown variables, even though this does not guarantee a unique solution.

4.3.2 Recovering the keystream

As the plaintext-ciphertext pairs are known by the attacker, thus the plain pixels $p_m(i,j)$ and cipher pixels $c_m^{(3)}(i,j)$ are known, for $0 \le i,j \le 1$. Let $\Delta c^{(r)}(i,j) = c_a^{(r)}(i,j) - c_b^{(r)}(i,j)$, which is the difference between a^{th} and b^{th} images for r-round output pixels at position (i,j).

Before recovering the keystream, we first determine the permutation keys (s_1,s_2) of the encryption scheme. It can be done by considering two plaintext-ciphertext pairs, i.e. an all black 2×2 plain image $\mathbf{P} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$ and another plain image $\mathbf{Q} = \begin{bmatrix} 128 & 0 \\ 0 & 128 \end{bmatrix}$, in which their sum1 and sum2 in Equations (4.3) and (4.4) are zero. Then, compute $\Delta c^{(3)}(i,j)$, for $0 \le i,j \le 1$. There are four possible cases for permutation keys and are listed in Table 4.1.

Table 4.1: List of possible cases for permutation keys s_1 and s_2

case	s_1	s_2	$\Delta c^{(3)}(0,0)$	$\Delta c^{(3)}(0,1)$	$\Delta c^{(3)}(1,0)$	$\Delta c^{(3)}(1,1)$
1	0	0	0	0	0	128
2	0	1	128	128	0	128
3	1	0	0	0	0	0
4	1	1	0	128	0	0

After knowing which case the permutation keys belong to, then the keystream $KS^{(1)} = (ks_1^{(1)}, ks_2^{(1)}, \ldots, ks_6^{(1)})$ and $KS^{(2)} = (ks_1^{(2)}, ks_2^{(2)}, \ldots,\ ks_6^{(2)})$ can be recovered by following the steps below.

1. Consider four 2×2 plaintext-ciphertext pairs $(\mathbf{P_1}, \mathbf{C_1}^{(3)})$, $(\mathbf{P_2}, \mathbf{C_2}^{(3)})$, $(\mathbf{P_3}, \mathbf{C_3}^{(3)})$, and $(\mathbf{P_4}, \mathbf{C_4}^{(3)})$, where the plain images are shown as follows.

$$\mathbf{P_1} = \begin{bmatrix} z & 0 \\ 0 & 0 \end{bmatrix}, \mathbf{P_2} = \begin{bmatrix} 0 & z \\ 0 & 0 \end{bmatrix}, \mathbf{P_3} = \begin{bmatrix} 0 & 0 \\ z & 0 \end{bmatrix} \text{ and } \mathbf{P_4} = \begin{bmatrix} 0 & 0 \\ 0 & z \end{bmatrix},$$

where $z \in \mathbb{Z}_{256}$.

2. There are 2^{32} possible candidates for $ks_5^{(1)}, ks_5^{(2)}, ks_6^{(1)}, ks_6^{(2)}$ in Equations (4.16)- (4.19), respectively. For each guess of 2^{32} possible values of the aforementioned four keys and $1 \leq m \leq 4$, compute

$$c_m^{(2)}(1-s_1, 1+s_2) = c_m^{(3)}(1,1) - c_m^{(3)}(0,0) - ks_6^{(2)}, \tag{4.20}$$

$$c_m^{(2)}(-s_1, 1+s_2) = c_m^{(3)}(1,0) - c_m^{(3)}(0,1) - 1 + ks_6^{(1)}[c_m^{(2)}(1-s_1, 1+s_2) + c_m^{(3)}(0,0)]^2, \tag{4.21}$$

$$c_m^{(2)}(1, s_2) + c_m^{(2)}(1,0) = c_m^{(3)}(0,1) - ks_5^{(2)} \tag{4.22}$$

$$c_m^{(2)}(0, s_2) + c_m^{(2)}(1,1) = c_m^{(3)}(0,0) - 1 + ks_5^{(1)}[c_m^{(2)}(1, s_2) + c_m^{(2)}(1,0)]^2. \tag{4.23}$$

3. To obtain $ks_2^{(1)}$, compute $\Delta c^{(2)}(0,1)$ for the images with $p(0, s_2) = z$ and $p(-s_1, 1+s_2) = z$, respectively, using Equation (4.13) as follows.

(a) For cases 1 and 3, $\Delta c^{(2)}(0,1) = 2\Delta c^{(1)}(1,0) = 2[-ks_2^{(1)}(2z+z^2)-z]$. Solve $ks_2^{(1)}$ by using $\Delta c^{(2)}(0,1)$ determined from Equations (4.21) and (4.20) for cases 1 and 3, respectively.

(b) For cases 2 and 4, $\Delta c^{(2)}(0,1) = \Delta c^{(1)}(1,1) + \Delta c^{(1)}(1,0) = z+[-ks_2^{(1)}(2z+z^2)-z]$. Solve $ks_2^{(1)}$ by using $\Delta c^{(2)}(0,1)$ determined from Equations (4.22) and (4.23).

4. Obtain $ks_1^{(1)}$ as follows.

(a) For cases 1 and 3, since $c^{(2)}(0,1)$ for image with $p(0,s_2)=z$ is obtained from Equations (4.21) and (4.20), respectively, substitute the known $c^{(2)}(0,1)$ and $ks_2^{(1)}$ into Equations (4.10) and (4.13) to obtain $2ks_1^{(2)}+ks_3^{(2)}$. Then, solve $ks_1^{(1)}$ by substituting the $2ks_1^{(2)}+ks_3^{(2)}$ and $ks_2^{(1)}$ into $c_m^{(2)}(0,1)$ for image with $p(1,s_2)=z$ in Equation (4.13).

(b) For cases 2 and 4, since $c^{(2)}(0,1)$ for image with $p(0,s_2)=z$ is obtained from Equation (4.22), substitute the known $c^{(2)}(0,1)$ and $ks_2^{(1)}$ into Equation (4.13) to obtain $ks_1^{(2)}+ks_2^{(2)}+ks_3^{(2)}$. Then, solve $ks_1^{(1)}$ by substituting the $ks_1^{(2)}+ks_2^{(2)}+ks_3^{(2)}$ and $ks_2^{(1)}$ into $c_m^{(2)}(0,1)$ for image with $p(1,s_2)=z$ in Equation (4.13).

5. From Steps 2 to 4, we have determined $\mathbf{C}_m^{(2)}$. From Equation (4.14), we have

$$\begin{aligned} ks_4^{(1)}(c_m^{(1)}(1-s_1,1+s_2)+c_m^{(2)}(0,0))^2 =& 1+c_m^{(1)}(-s_1,1+s_2)+c_m^{(2)}(0,1) \\ &-c_m^{(2)}(1,0), \end{aligned} \tag{4.24}$$

where $c^{(1)}(1-s_1,1+s_2)$ for $0 \leq s_1,s_2 \leq 1$ are

$$c_m^{(1)}(1-s_1,1+s_2)$$
$$= \begin{cases} c_m^{(1)}(1,1) = & p_m(1,1)+1-ks_1^{(1)}(p_m(1,0))^2+p_m(0,0)+ks_2^{(2)}, \quad \text{for case 1,} \\ c_m^{(1)}(1,0) = & 1-ks_2^{(1)}(p_m(1,0)+c_m^{(1)}(0,0))^2+p_m(0,0)+ \\ & p_m(1,1)+ks_1^{(2)}, \quad \text{for case 2,} \\ c_m^{(1)}(0,1) = & p_m(1,0)+ks_1^{(2)}, \quad \text{for case 3,} \\ c_m^{(1)}(0,0) = & 1-ks_1^{(1)}(p_m(1,1))^2+p_m(0,1), \quad \text{for case 4.} \end{cases} \tag{4.25}$$

(a) For the first three cases, form three equations from Equation (4.24) with different m. Based on the secret key involved in $c_m^{(1)}(1-s_1,1+s_2)$ of Equation (4.25), we can solve the three equations simultaneously to obtain the following secret keys.

i. For case 1, we can obtain $ks_4^{(1)}$ and $ks_2^{(2)}$. Since $c_m^{(1)}(-s_1,1+s_2)$ in Equation (4.24) is $c_m^{(1)}(0,1) = p_m(1,1)+ks_1^{(2)}$, therefore $ks_1^{(2)}$ can be obtained by substituting $ks_4^{(1)}$ and $ks_2^{(2)}$ into Equation (4.24).

ii. For cases 2 and 3, we can obtain $ks_4^{(1)}$ and $ks_1^{(2)}$.

(b) For case 4, from Equation (4.25), no secret key involves in $c_m^{(1)}(0,0)$, therefore only two equations from (4.24) are needed to determine $ks_4^{(1)}$ and $ks_1^{(2)}$ with different m.

6. From Steps 4 and 5, we can obtain the following secret keys.

(a) For cases 1 and 3, obtain $ks_3^{(2)}$ by substituting $ks_1^{(2)}$ obtained from Step 5(a)i into $2ks_1^{(2)}+ks_3^{(2)}$ from Step 4a.

(b) For cases 2 and 4, obtain $ks_2^{(2)}+ks_3^{(2)}$ by substituting $ks_1^{(2)}$ obtained from Step 5(a)ii or Step 5b into $ks_1^{(2)}+ks_2^{(2)}+ks_3^{(2)}$ from Step 4b.

7. Since $c_m^{(1)}(1-s_1,1+s_2)$ from Equation (4.25) can been recovered based on the $ks_1^{(2)}$ and $ks_2^{(2)}$ determined in Step 5, therefore we can obtain $ks_4^{(2)}$ by

substituting $c_m^{(1)}(1-s_1,1+s_2)$ into Equation (4.15) as follows.

$$ks_4^{(2)} = c_m^{(2)}(1,1) - c_m^{(2)}(0,0) - c_m^{(1)}(1-s_1,1+s_2). \tag{4.26}$$

8. The remaining subkeys can be obtained by solving Equation (4.12) as follows.
   (a) For case 1, since $ks_2^{(1)}$, $ks_1^{(2)}$ and $ks_2^{(2)}$ have been determined in Steps 3a and 5(a)i, therefore $ks_3^{(1)}$ can be solved by substituting the values in (4.12) with any m value. Take $m=1$ as example, $ks_3^{(1)}$ can be determined by solving $4ks_3^{(1)}(1-4ks_2^{(1)}+ks_1^{(2)})^2 = 5 - c_1^{(2)}(0,0) + ks_2^{(2)}$.
   (b) For case 2, since $ks_2^{(2)}$ is unknown, therefore form three equations from (4.12) with different m. Substitute $ks_2^{(1)}$ and $ks_1^{(2)}$ that obtained in Steps 3b and 5(a)ii into these three equations. Then, solve the equations simultaneously to get $ks_3^{(1)}$ and $ks_2^{(2)}$. Obtain $ks_3^{(2)}$ by substituting $ks_2^{(2)}$ into $ks_2^{(2)}+ks_3^{(2)}$ that obtained in Step 6b.
   (c) For case 3, take two equations from Equation (4.12) with different m. Substitute $ks_2^{(1)}$ and $ks_1^{(2)}$ obtained in Steps 3a and 5(a)ii into the equations. Obtain $ks_3^{(1)}$ and $ks_2^{(2)}$ by solving these two equations simultaneously.
   (d) For case 4, it is similar to Step 8b. Substitute $ks_2^{(1)}$ and $ks_1^{(2)}$ obtained from Steps 3b and 5*b* into three equations formed by using Equation (4.12) with different m. Solve the equations simultaneously to obtain $ks_3^{(1)}$ and $ks_2^{(2)}$. Then, $ks_2^{(2)}$ is substituted into $ks_2^{(2)}+ks_3^{(2)}$ determined in Step 6b to obtain $ks_3^{(2)}$.

We demonstrate the algorithm of recovery process for the permutation keys $s_1=0$ and $s_2=0$ under case 1, using $\mathbf{P_m}$, for $1 \le m \le 4$ with $z=1$ in Algorithm 2.

Analysis. To test the performance of the proposed recovering method, simulations are conducted using Matlab R2019a, running on a personal computer with Intel(R) Core(TM) i5-8250 CPU @ 1.60GHz, 8 GB memory and a Windows 10 operating system. The keystream generated by using secrets keys ($s_1 = 45$, $s_2 = 170$, $x_0 = 0.213377264386424$ and $y_0 = 0.166908249009117$), which are the same as provided in Ping et al. (2018), have been identified for plaintext-ciphertext pairs with $z = 1$ and $z = 2$ in Step 1. The possible values for keystreams ($KS^{(1)}$, $KS^{(2)}$) and ($KS^{(3)}$, $KS^{(4)}$) are 2^{51} and 2^{60}, respectively.

To validate the results for worst case scenario, the possible values of the $(ks_1^{(1)}, ks_2^{(1)}, \ldots, ks_6^{(1)})$ and $(ks_1^{(2)}, ks_2^{(2)}, \ldots, ks_6^{(2)})$ for $z = 1$ have also been verified on another 120 different sets of keystreams using Matlab environment.

Let $|u|$ denote the possible value of u. Recall that case 1 is $s_1 = 0, s_2 = 0$, case 2 is $s_1 = 0, s_2 = 1$, case 3 is $s_1 = 1, s_2 = 0$, and case 4 is $s_1 = 1, s_2 = 1$. The results of possible values are summarized as follows.

i. From Step 2, we have $\prod_{k=5}^{6} |ks_k^{(1)}| \cdot |ks_k^{(2)}| = (256)^4 = 2^{32}$.

ii. From Step 3, for cases 1 and 3, $|ks_2^{(1)}| = 2$, while for cases 2 and 4, $|ks_2^{(1)}| = 1$.

iii. From Step 4, the possible value of $ks_1^{(1)}$ can be found as follows.

$$\begin{aligned} &\text{For case 1,} \quad |ks_1^{(1)}| \in (8, 2^6); \quad \text{For case 3,} \quad |ks_1^{(1)}| \in (8, 2^8); \\ &\text{For case 2,} \quad |ks_1^{(1)}| \in (1, 2); \quad \text{For case 4,} \quad |ks_1^{(1)}| \in (1, 2). \end{aligned} \tag{4.27}$$

iv. From Step 5, the possible value for $ks_4^{(1)}$ and $ks_1^{(2)}$ or $ks_2^{(2)}$ are determined together as follows.

$$\text{For case 1, } |ks_4^{(1)}| \in (1,2), |ks_2^{(2)}| \in (4, 2^8) \text{ and } |ks_1^{(2)}| \in (1,2); \tag{4.28}$$

$$\text{For case 2, } |ks_4^{(1)}| \cdot |ks_1^{(2)}| \in (4, 2^{10}); \tag{4.29}$$

$$\text{For case 3, } |ks_4^{(1)}| \in (1,2) \text{ and } |ks_1^{(2)}| \in (4, 2^8); \tag{4.30}$$

$$\text{For case 4, } |ks_4^{(1)}| = |ks_1^{(2)}| = 1. \tag{4.31}$$

v. From Steps 6 and 7, the possible values for $ks_4^{(2)}$ and $ks_3^{(2)}$ are shown as follows.

$$\text{For cases 1 and 3, } |ks_3^{(2)}| = |ks_4^{(2)}| = 1; \tag{4.32}$$

$$\text{For cases 2 and 4, } |ks_4^{(2)}| = 1. \tag{4.33}$$

vi. From Step 8, the possible values for the remaining keys are obtained as follows.

$$\text{For case 1, } s_1 = 0, s_2 = 0,\ |ks_3^{(1)}| \in (1, 2^8); \tag{4.34}$$

$$\text{For case 2, } s_1 = 0, s_2 = 1,\ |ks_3^{(1)}| \cdot |ks_2^{(2)}| \in (1, 2^6) \text{ and } |ks_3^{(2)}| = 1; \tag{4.35}$$

$$\text{For case 3, } s_1 = 1, s_2 = 0,\ |ks_3^{(1)}| \in (4, 2^8) \text{ and } |ks_2^{(2)}| = 1; \tag{4.36}$$

$$\text{For case 4, } s_1 = 1, s_2 = 1,\ |ks_3^{(1)}| \cdot |ks_2^{(2)}| \in (1, 2^6) \text{ and } |ks_3^{(2)}| = 1. \tag{4.37}$$

Algorithm 2: Chosen Plaintext Attack

```
Input: Chosen plaintext-ciphertext pairs $(\mathbf{P_1}, \mathbf{C_1}^{(3)})$, $(\mathbf{P_2}, \mathbf{C_2}^{(3)})$, $(\mathbf{P_3}, \mathbf{C_3}^{(3)})$, and $(\mathbf{P_4}, \mathbf{C_4}^{(3)})$
Output: Keystreams $KS^{(1)}$ and $KS^{(2)}$
for $ks_6^{(2)} = 0:255$ do
  for $ks_6^{(1)} = 0:255$ do
    for $ks_5^{(2)} = 0:255$ do
      for $ks_5^{(1)} = 0:255$ do
        for $m = 1:4$ do
          $c_m^{(2)}(1,1) = c_m^{(3)}(1,1) - c_m^{(3)} - ks_6^{(2)}$ ;    // Refer to Eq. (4.20)
          $c_m^{(2)}(0,1) = c_m^{(3)}(1,0) - c_m^{(3)}(0,1) - 1 + ks_6^{(1)}[c_m^{(2)}(1,1) + c_m^{(3)}(0,0)]^2$ ;    // Refer to Eq. (4.21)
          $c_m^{(2)}(1,0) = 1/2(c_m^{(3)}(0,1) - ks_5^{(2)})$ ;    // Refer to Eq. (4.22)
          $c_m^{(2)}(0,0) = c_m^{(3)}(0,0) - 1 + ks_5^{(1)}[c_m^{(2)}(1,0) + c_m^{(2)}(1,0)]^2 - c_m^{(2)}(1,1).$ ;    // Refer to Eq. (4.23)
          $ks_2^{(1)} = []; ks_1^{(2)} = []$
          $\Delta c^{(2)}(0,1) = c_1^{(2)}(0,1) - c_4^{(2)}(0,1)$
          for $k = 0:255$ do
            if $\Delta c^{(2)}(0,1) == 2[-k(3)^2 - 1]$ then
              $ks_2^{(1)} = [ks_2^{(1)}; k]$ ;    // Refer to Step 3a
              $2ks_1^{(2)} + ks_3^{(2)} = c_1^{(2)}(1,0) - 2 + 2k$ ;    // Refer to Step 4a
              for $l = 0:255$ do
                if $2[1 - k(1-l)^2] == c_3^{(2)}(0,1) - (2ks_1^{(2)} + ks_3^{(2)})$ then
                  $ks_1^{(1)} = [ks_1^{(1)}; l]$
                  $ks_4^{(1)} = []; ks_2^{(2)} = []; ks_1^{(2)} = []; ks_4^{(2)} = [];$
                  for $s = 0:255$ do
                    for $t = 0:255$ do
                      if $s[2 + t + c_1^{(2)}(0,0)]^2 - s[2 + t + c_2^{(2)}(0,0)]^2 == \Delta c^{(2)}(0,1) - \Delta c^{(2)}(1,0)$ ;  // Refer to Eq. (4.24)
                      then
                        $ks_4^{(1)} = [ks_4^{(1)}; s]$
                        $ks_2^{(2)} = [ks_2^{(2)}; t]$
                        for $u = 0:255$ do
                          $u = s[2 + t + c_3^{(2)}(0,0)]^2 - 1 - c_3^{(2)}(0,1) + c_3^{(2)}(1,0)$
                          $ks_1^{(2)} = [ks_1(2); u]$ ;    // Refer to Step 5(a)i
                          $ks_3^{(2)} = c_1^{(2)}(1,0) - 2 + 2k - 2ks_1^{(2)}$ ;  // Refer to Step 6a
                          $c_1^{(1)}(1,1) = 2 + t$ ;    // Refer to Eq. (4.25)
                          $v = c_1^{(2)}(1,1) - c_1^{(2)}(0,0) - c_1^{(1)}(1,1);$  // Refer to Step 7 Eq. (4.26)
                          $ks_3^{(1)} = [];$
                          for $w = 0:255$ do
                            if $4w(1 - 4k + u)^2 = 5 - c_1^{(2)}(0,0) + t$ then
                              $ks_3^{(1)} = [ks_3^{(1)}; w]$ ; // Refer to Step 8a
                            end
                          end
                        end
                      end
                    end
                  end
                end
              end
            end
          end
        end
      end
    end
  end
end
```

From the results above, we found that case 3 is the worst case scenario that having the highest possible value of secret keys. The total possible value of $(ks_1^{(1)}, ks_2^{(1)}, \ldots, ks_6^{(1)})$ and $(ks_1^{(2)}, ks_2^{(2)}, \ldots, ks_6^{(2)})$ for this case is $2^{32} \times 2 \times 2^8 \times 2 \times 2^8 \times 1 \times 2^8 \times 1 = 2^{58}$.

The time complexity of the recovery process under the worst case scenario are calculated as follows. To recover permutation keys s_1 and s_2, it requires two plaintext-ciphertext pairs. Step 1 involves 4 equations. For Step 2, with 2^8 possible values of $ks_6^{(2)}$, Equation (4.20) is required to run for $2^8 \times 4 = 2^{10}$ times to get $c_m^{(2)}(1-s_1, 1+s_2)$. With 2^{16} possible values for $c_m^{(2)}(1-s_1, 1+s_2)$ and $ks_6^{(1)}$, Equation (4.21) is required to run for $2^{16} \times 4 = 2^{18}$ times. Similarly, both $ks_5^{(2)}$ and $ks_5^{(1)}$ have 2^8 possible values. Hence, Equation (4.22) runs for $2^{16} \times 2^8 \times 4 = 2^{26}$ times to get $c_m^{(2)}(1, s_2) + c_m^{(2)}(1, 0)$ while Equation (4.23) runs for $2^{24} \times 2^8 \times 4 = 2^{34}$ times to get $c_m^{(2)}(0, s_2) + c_m^{(2)}(1, 1)$.

With 2^{32} possible values for $ks_k^{(1)}$ and $ks_k^{(2)}$ for $k = 5, 6$, the remaining secret keys can be obtained as follows. To obtain $ks_2^{(1)}$, Step 3 requires to compute $\Delta c^{(2)}(0, 1)$ for 2^{32} times. Since there are 2 possible values of $ks_2^{(1)}$ under the worst case, Step 4 computes $2^{32} \times 2 \times 2 = 2^{34}$ equations to obtain $ks_1^{(1)}$. Since there are 2^8 possible value for $ks_1^{(1)}$, Step 5 requires approximately $2^{33} \times 2^8 \times 6 \approx 2^{38.58}$ equations to obtain $ks_4^{(1)}$ and $ks_1^{(2)}$ or $ks_2^{(2)}$. There are $2 \times 2^8 \times 1 = 2^9$ possible values for $ks_4^{(1)}$ and $ks_1^{(2)}$ or $ks_2^{(2)}$. So, Steps 6 and 7 require $2^{41} \times 2^9 \times 1 = 2^{50}$ equations for each step. Finally, Step 8 requires approximately $2^{50} \times 6 \times 1 \approx 2^{52.58}$ equations to obtain $ks_3^{(1)}$ or $ks_2^{(2)}$ and $ks_3^{(2)}$.

The whole process involves approximately $2^2 + 2^{10} + 2^{18} + 2^{26} + 2^{34} + 2^{32} + 2^{34} + 2^{38.58} + 2^{50} \times 2 + 2^{52.58} \approx 2^{53}$ equations in total. Since 3-round encryption involves 12 equations, so the proposed attack has the time complexity around $2^{53}/12 \approx 2^{49.41}$ encryptions to reduce 2^{96} possible values of $(ks_1^{(1)}, ks_2^{(1)}, \ldots, ks_6^{(1)})$ and $(ks_1^{(2)}, ks_2^{(2)}, \ldots, ks_6^{(2)})$ to 2^{58} possible values under the worst case scenario for $z = 1$.

4.3.3 Recovering the secret keys

Recall from Subsection 4.3.2, let the four plain images in Step 1 have $z = 1$. The recovered keystream are needed to recover updated initial conditions x_0' and y_0'. To recover the correct secret keys $\langle x_0, y_0, s_1, s_2\rangle$, we require another four chosen plaintext-ciphertext pairs $\mathbf{P_m}$, for $5 \leq m \leq 8$, to generate another set of keystreams. Their updated initial conditions calculated from Equations (4.6) and (4.7) are x_0'' and y_0'' and the subkeys generated by Equation (4.1) are denoted as $ks_i^{(3)}$ and $ks_i^{(4)}$, for $1 \leq i \leq 6$. For ease of understanding, we let $d = 0$ in Equation (4.1) as it is used to avoid harmful transient effect and will not affect the validity of our proposed attack. We perform the following procedure to recover the secret parameters.

1. Repeat Steps 1 to 8 in Subsection 4.3.2 to obtain $ks_i^{(3)}$ and $ks_i^{(4)}$, for $1 \leq i \leq 6$, with $z = 2$ for the four plain images in Step 1.

2. Guess the value for (x_0', y_0') with 10^{15} computational precision of floating numbers and perform the following steps.

   (a) For $i = 1$, use the guess value of (x_0', y_0') to compute

   $$\begin{aligned} x_i' &= 1 - a(x_{i-1}')^2 + y_{i-1}', \\ y_i' &= b x_{i-1}'. \end{aligned}$$

   If $\text{floor}(|x_i'| \times 10^{15}) (\text{mod } 256) = ks_i^{(1)}$ and $\text{floor}(|y_i'| \times 10^{15}) (\text{mod} 256) = ks_i^{(2)}$, then store the guess value of (x_0', y_0').

   (b) Repeat Step 2a by using the corresponding $ks_i^{(1)}$ and $ks_i^{(2)}$, for $i > 1$ to reduce the number of possible candidates of (x_0', y_0').

3. From Equations (4.6) and (4.7), the updated initial values are different from

6 decimal places onwards. Repeat Step 2 to get the possible (x_0'', y_0'') by using $ks_i^{(3)}$ and $ks_i^{(4)}$, for $1 \leq i \leq 6$.

4. Compute $d_1 = |x_0'' - x_0'|$.

   (a) If $d_1 < 10^{-6}$, then Equation (4.6) becomes

$$x_0' \quad = x_0 \pm \sigma_1', \quad x_0'' \quad = x_0 \pm \sigma_1''. \tag{4.38}$$

   where $\sigma_1' = \frac{(1+1)\times(1+1)}{257^2} \times \frac{s_1}{s_2}$ and $\sigma_1'' = \frac{(2+1)\times(4+1)}{257^2} \times \frac{s_1}{s_2}$.
   Then, compute $\frac{s_1}{s_2} = \frac{d_1}{\frac{15}{257^2} - \frac{4}{257^2}}$.

   (b) Else, we have

$$x_0' \quad = x_0 \pm [\sigma_1' \times 10^5 - \text{floor}(\sigma_1' \times 10^5)] \times 10^{-5}, \tag{4.39}$$

$$x_0'' \quad = x_0 \pm [\sigma_1'' \times 10^5 - \text{floor}(\sigma_1'' \times 10^5)] \times 10^{-5}. \tag{4.40}$$

   - Note that the upper bound of σ_1' and σ_1'' are $\frac{4}{257^2} = 0.000060561$ and $\frac{15}{257^2} = 0.000227104$, respectively, when $\frac{s_1}{s_2} = 1$.
   - The upper bound of $[\text{floor}(\sigma_1' \times 10^5)] \times 10^{-5} = 0.00006$.
   - The upper bound of $[\text{floor}(\sigma_1'' \times 10^5)] \times 10^{-5} = 0.00022$.
   - So, the possible value of their difference $d_2 = [\text{floor}(\sigma_1'' \times 10^5) - \text{floor}(\sigma_1' \times 10^5)] \times 10^{-5} \in \{0, 0.00001, 0.00002, \ldots, 0.00016\}$.

   Then, compute $\frac{s_1}{s_2} = \frac{d_1 + d_2}{\frac{15}{257^2} - \frac{4}{257^2}}$ for all the possible d_2.

5. Obtain x_0 by substituting $\frac{s_1}{s_2}$ from Step 4 into Equations (4.38) if or (4.39). Similarly, obtain y_0 by substituting $\frac{s_1}{s_2}$ into the following equation.

$$\sigma_2' = \frac{(1+2)\times(1+2)}{258^2} \times \frac{s_1}{s_2}, \tag{4.41}$$

$$y_0' = y_0 \pm \sigma_2', \qquad \text{if } d_2 < 10^{-6}; \tag{4.42}$$

$$y_0' = y_0 \pm [\sigma_2' \times 10^5 - \text{floor}(\sigma_2' \times 10^5)] \times 10^{-5}, \qquad \text{if } d_2 \geq 10^{-6}. \tag{4.43}$$

Analysis. Recall the keystream recovery process in Subsection 4.3.2, there are 2^{58} possible values of $KS^{(1)}$ $(ks_1^{(1)}, ks_2^{(1)}, \ldots, ks_6^{(1)})$ and $KS^{(2)} = (ks_1^{(2)}, ks_2^{(2)}, \ldots, ks_6^{(2)})$ under the worst case scenario for plain image with $z = 1$. For each possible value of $KS^{(1)}$ and $KS^{(2)}$, there are $(\frac{10^{15}}{2^8})^2 \approx 2^{83.66}$ possible candidates for x_0' and y_0' in Step 2. After repeating Step 2b using $ks_i^{(1)}$ and $ks_i^{(2)}$, for $2 \leq i \leq 6$, the number of candidates for x_0' and y_0' are reducing approximately to $\frac{10^{15\times 2}}{2^{8\times 12}} \approx 2^{3.66}$. [1] In Step 4, we recover $\frac{s_1}{s_2}$ which is the equivalent key for s_1 and s_2, meaning that the 2^{256} possible values of s_1 and s_2 are reduced to 17 possible values of $\frac{s_1}{s_2}$. Since this attack requires four additional plaintext-ciphertext pairs, therefore the time complexity for the whole secret keys processes is $2 \times 2^{49.41} \times 10^{15+15+10+10} \approx 2^{216.51}$. According to Yap et al. (2016), an encryption scheme is insecure if its time complexity is less than $2^{|K|}$ encryptions, where $|K|$ denotes the length of the secret key K in bits. We show that the time complexity of the recovery process is much lower than the length of the secret key claimed by Ping et al., around 2^{356}. Through the key recovery process, the possible value of x_0, y_0 and $\frac{s_1}{s_2}$ has been reduced to $2^{58} \times 2^{3.66} \times 17 \approx 2^{65.74}$, then the recovered x_0, y_0 and $\frac{s_1}{s_2}$ can be used to recover other images of various sizes.

4.4 Investigating the efficiency of two-point diffusion strategy

Ping et al. has pointed out that the two-point diffusion strategy can enhance diffusion effect if more than one processing unit is used. However, the proposed method encrypts every two image pixels sequentially and the value of the current cipher pixel depends on its previous cipher pixels. As shown in Figure 4.4, to diffuse the last pixels 7 and 8, all the previous pixels need to be diffused

[1] This result is estimated by using x_0' and y_0' with the computational precision of floating point from 10^3 to 10^7.

first. This is just similar to the traditional diffusion method, whereby each pixel is influenced by one or more previous pixels. This can greatly jeopardize the parallelism of algorithm and its efficiency.

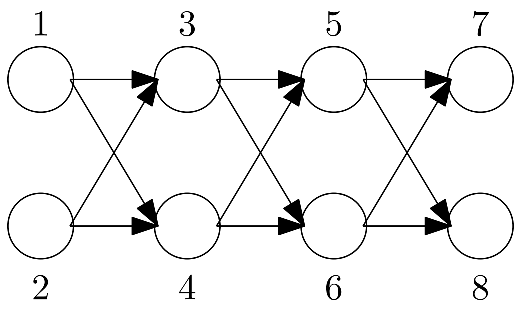


Figure 4.4: Two-diffusion strategy proposed by Ping et al.

To allow the algorithm to be executed parallelly, the image needs to be divided into block and running on the multi-core processing units simultaneously. However, the processing unit that processes the image data of the current block needs to wait for the other processing unit that processes the previous block to complete. This time-consuming encryption method is similar to encryption algorithms with ciphertext block chain (CBC) mode, whereby one cipher pixel is affected by previous cipher (Wang et al., 2018). This shortage limits the application on the platform based on field programmable gate array (FPGA)/complex programmable logic device (CPLD) or digital circuits that can support parallel processing.

Since computational speed can be greatly accelerated in the parallel encryption mode, the CBC-like diffusion mode must be avoided. To have the parallel encryption mode, there are suggestions to be considered.

1. The image data is divided into blocks.
2. Each processing unit processes each block independently and possesses their own memory.
3. The encryption within each block must satisfy the confusion and diffusion properties adequately (Katz and Lindell, 2020). This can be

done by using the block cipher with counter (CTR) mode to avoid the current plaintext block from being influenced by previous block. This method is suitable for parallel computing because the counter values are encrypted independently in parallel before combining with the plaintext to produce the ciphertext. Besides, it is also proven that the CTR mode is secure against the chosen plaintext attack.

4. To further enhance the diffusion effect, the encrypted data are exchanged and communicated among the processing units.

4.5 Summary

In this chapter, we analyzed an image encryption network that adopting two point diffusion strategy, where its diffusion process is intermingled with the permutation process. The analysis shows that chosen plaintext attack can be launched against the image encryption scheme by using ten 2×2 plain images. The time complexity of the attack is $2^{216.51}$ encryptions, which is much lower than the key space claimed by Ping et al., i.e. 2^{356}. The possible values of secret keys x_0 and y_0 and the equivalent key $\frac{s_1}{s_2}$ has also been reduced substantially to $2^{65.75}$. The recovered keys can be used to recover the plain image with larger size. The key space can be further diminished when pseudorandom sequences generated by the Henon map are implemented using digital computers in real applications, with a more detailed discussion provided in Chapter 6.

Additionally, the method proposed by Ping et al. does not achieve a significant improvement in efficiency compared to conventional diffusion methods, primarily because the encryption process is not parallelized. To ensure parallelism during the encryption process, images should be classified into different levels based on the available parallel resources of the computing system. The diffusion should then be performed within each group under a parallelism framework. To avoid sequential encryption within the group, using a block cipher based on counter mode is a popular method for enabling parallel encryption.

CHAPTER 5

DYNAMICS ANALYSIS OF TWO ONE-DIMENSIONAL CHAOTIC MAPS

This chapter explores the dynamics of two one-dimensional chaotic maps with the aim of proposing new chaotic map through cascading methods to enhance their chaotic behavior. This study is significant because the fundamental properties of chaos, such as sensitivity to initial conditions, unpredictability, and randomness, which are essential for designing secure encryption algorithms. By understanding and improving these properties, we intend to contribute to the development of more robust image encryption schemes.

In this chapter, we focus on the characteristics of one-dimensional chaotic maps, which are crucial for achieving the second research objective of this thesis. We have selected two well-known chaotic maps, namely the Logistic map and the Beta map, to compare the performance of our proposed map with those existing in the literature. While these maps are widely used, they have certain limitations, for example, the Logistic map contains periodic windows within its chaotic range, which can reduce its overall chaoticity and introduce potential vulnerabilities in encryption applications.

By examining these limitations, this chapter not only highlights the weaknesses of commonly used chaotic maps but also underscores the importance of enhanced chaotic behavior for image encryption schemes. The findings presented here establish a foundation for developing improved chaotic maps through cascading methods.

5.1 Introduction

Over the past decade, chaotic systems have received attentions from many researchers to study their chaotic behaviors. This is due to the interesting characteristics of chaotic systems, for example, aperiodicity, high sensitivity to the initial conditions and system parameters, ergodicity and random-like behaviors. This is just analogous to the confusion and diffusion properties of cryptographic properties (Shannon, 1949). Matthews (1989) was the first person to apply chaotic system to image encryption technology. Since then, the popularity of using chaos in cryptography has been grew significantly.

Chaotic system has been widely applied in designing image encryption scheme. This is because the conventional encryption methods such as Data Encryption Standard (DES) (National Bureau of Standards, 1977), Advanced Encryption Standard (AES) (Daemen and Rijmen, 2013), and International Data Encryption Algorithm (IDEA) (Lai and Massey, 1990) are no longer suitable to encrypt image data because of the bulky data capacity and high correlation among the pixels. Chaotic map is therefore applied in (a) constructing permutation matrices in the encryption process; (b) generating a chaotic pseudorandom sequences; and (c) producing the ciphertext by having the plain pixel to be the secret keys and the chaotic map to be the encryption operation (Zhang et al., 2012).

Wu et al. (2018b) designed an image encryption based on a chaotic map which is formed by combining 2D-Henon map and a Sine map. The authors used the chaotic map to generate keystream and then apply DNA approach to encrypt the plain image. An image encryption scheme designed based on 2D Logistic-Sine-Cosine map was presented by Huang (2019). The chaotic system are created based on 2D Logistic, Sine and Cosine maps. Zhu et al. (2019) presented a new chaotic map based on 2D Logistic-Modulated-Sine-

Coupling-Logistic map for image encryption, whereby Sine map is modulated by the Logistic map and then the result of modulation and Sine map are coupled together.

In this chapter, we introduce a new chaotic map, called Logistic-Beta map which is formed by combining Logistic map with Beta map. Logistic map is a one-dimensional map which has been widely used in encryption scheme (May, 1976). Beta map is a chaotic map proposed by Zahmoul et al. (2017), which is based on a statistical distribution, called Beta function. We study the chaotic behaviors of the Logistic-Beta map, i.e. its trajectory, bifurcation diagram and Lyapunov exponent. We also demonstrate our proposed chaotic map has a better chaotic behaviors than the classical Logistic and Beta maps, and a one-dimensional logistic-based chaotic map.

Organization: The remainder of this chapter is organised as follows. Section 5.2 describes the preliminaries of some exisiting chaotic maps. Section 5.3 presents the mathematical model of our newly proposed chaotic map. Section 5.4 discusses the dynamical analysis of our proposed map. The last section summarizes the chapter.

5.2 Preliminaries

This section briefly discusses the Logistic map and Beta map which are going to generate our proposed chaotic map. We also discuss a one-dimensional chaotic map that designed based on Logistic map. We will compare the chaotic behaviors of our proposed chaotic map with the following three chaotic maps in next section.

Logistic map

Logistic map is a one-dimensional discrete-time dynamical system proposed by May (1976). It is an iterated map that represented by a first order difference equations as

$$x_{n+1} = rx_n(1-x_n), \tag{5.1}$$

where $x_n \in (0,1)$ and $r \in [0,4]$.

Beta map

Zahmoul et al. (2017) proposed a chaotic map based on a Beta function, known as Beta map. It is defined as

$$y_{n+1} = \mu \cdot B(y_n; y_1, y_2, c, d), \tag{5.2}$$

where $B(y_n; y_1, y_2, c, d)$ denotes the Beta function for $y = \{y_n\}_{n=0}^{\infty}$ and μ is a multiplier that controls the amplitude of Beta map. The beta function of y is represented by

$$B(y; y_1, y_2, c, d) \quad = \begin{cases} \left(\frac{y-y_1}{y_m-y_1}\right)^c \left(\frac{y_2-y}{y_2-y_m}\right)^d, & \text{if } y \in (y_1, y_2); \\ 0, & \text{otherwise.} \end{cases} \tag{5.3}$$

Given that $y_m = \frac{cy_2+dy_1}{c+d}$ denotes the weighted mean of y_1 and y_2, where $c, d, y_1, y_2 \in \mathbb{R}$ and $y_1 < y_2$. The parameters c and d are determined as

$$c \quad = p_1 + q_1 \times e; \tag{5.4}$$

$$d \quad = p_2 + q_2 \times e, \tag{5.5}$$

where e is a bifurcation parameter and p_1, p_2, q_1 and $q_2 \in \mathbb{R}$ are randomly chosen constants.

A chaotic map must be bounded. To prove this, we identify the value of μ that results in $y_{n+1} \in (y_1, y_2)$. We know that the first derivative test can help to find the minima and maxima of a function, then we compute $\frac{dy_{n+1}}{dy_n} = 0$ by fixing y_1, y_2, y_m as constant.

$$\begin{aligned}
\tfrac{dy_{n+1}}{dy_n} = {} & \mu\Bigg[d\Big(\frac{y_n - y_1}{y_m - y_1}\Big)^c\Big(\frac{y_2 - y_n}{y_2 - y_m}\Big)^{d-1}\Big(-\frac{1}{y_2 - y_m}\Big) + \\
& c\Big(\frac{y_2 - y_n}{y_2 - y_m}\Big)^d\Big(\frac{y_n - y_1}{y_m - y_1}\Big)^{c-1}\Big(\frac{1}{y_m - y_1}\Big)\Bigg] \\
0 = {} & \mu \cdot \frac{(y_n - y_1)^{c-1}(y_2 - y_n)^{d-1}}{(y_m - y_1)^c(y_2 - y_m)^d} \cdot [c(y_2 - y_n) - d(y_n - y_1)] \\
y_n = {} & \frac{cy_2 + dy_1}{c + d}.
\end{aligned} \tag{5.6}$$

Noted that $y_n = y_m$. Next, we compute the second derivative on y_{n+1} with respect to y_n as

$$\begin{aligned}
\frac{d^2 y_{n+1}}{dy_n^2} = {} & \mu\big(\tfrac{y_n - y_1}{y_m - y_1}\big)^{c-2}\big(\tfrac{y_2 - y_n}{y_2 - y_m}\big)^{d-2}\big(\tfrac{1}{(y_2 - y_m)(y_m - y_1)}\big)^2 \cdot \\
& \big(d(y_n - y_1)[-c(y_2 - y_n) + (d-1)(y_n - y_1)] + \\
& c(y_2 - y_n)[-d(y_n - y_1) + (c-1)(y_2 - y_n)]\big)
\end{aligned} \tag{5.7}$$

Then, substitute Equation (5.6) into Equation (5.7).

$$\begin{aligned}
\left.\frac{d^2 y_{n+1}}{dy_n^2}\right|_{y_n = y_m} = {} & \mu\big(\tfrac{1}{(y_2 - y_m)(y_m - y_1)}\big)^2 \cdot \big[cd(y_m - y_1)(y_1 - y_2) \\
& + cd(y_2 - y_m)(y_1 - y2)\big] < 0, \quad \because y_1 < y_2.
\end{aligned} \tag{5.8}$$

Therefore, y_m is the local maximum.

By letting $y_n = y_m$, we determine the range of μ by substituting Equation (5.6) into Equations (5.2) and (5.3) as

$$y_1 < \mu \cdot \big(\tfrac{y_m - y_1}{y_m - y_1}\big)^c\big(\tfrac{y_2 - y_m}{y_2 - y_m}\big)^d < y_2. \tag{5.9}$$

Hence, $\mu \in (y_1, y_2)$.

Modified Logistic map

The is a chaotic map designed by modifying the Logistic map discussed in Equation (5.1). It is proposed by Lestari et al. (2018) to allow the initial values to be positive or negative. It can be defined by

$$x_{n+1} = \begin{cases} g_1(x_n); \\ h_1(x_n), \end{cases} \tag{5.10}$$

for $x_n \in (-1, 1)$.

The recursive equation of the modification is given as

$$x_{n+1} = \begin{cases} (-\frac{3}{2}|r| - \sqrt{2}|r|) \cdot x_n \cdot ((2\sqrt{2} - 2)x_n + 1), & \text{for } -1 < x_n < 0; \\ (-\frac{3}{2}|r| - \sqrt{2}|r|) \cdot x_n \cdot ((2\sqrt{2} - 2)x_n - 1), & \text{for } 0 \leq x_n < 1, \end{cases} \tag{5.11}$$

where $r \in [-4, 4]$. In Section 5.4, we compare the dynamical performance of this chaotic map with our proposed map.

5.3 The proposed chaotic map

The newly proposed chaotic map, called Logistic-Beta map is designed by combining of two chaotic maps, i.e. Logistic map and Beta map. The mathematical model of our new one-dimensional chaotic map is represented by

$$x_{n+1} = f(x_n) = g(h(x_n)), \text{and } f : [0,1] \to [0,1],$$

where $h(\cdot)$ represents the Beta map with $\mu = 1$ given in Equation (5.2). Beta map is chosen to enlarge phase space, while $g(\cdot)$ represents the Logistic map given in Equation (5.1). Therefore, the Logistic-Beta map is defined as

$$x_{n+1} = r\left(\frac{x_n - y_1}{y_m - y_1}\right)^c \left(\frac{y_2 - x_n}{y_2 - y_m}\right)^d \left[1 - \left(\frac{x_n - y_1}{y_m - y_1}\right)^c \left(\frac{y_2 - x_n}{y_2 - y_m}\right)^d\right], \tag{5.12}$$

where n is the iteration number, $y_m = \frac{cy_2 + dy_1}{c+d}$ and $c, d, y_1, y_2 \in \mathbb{R}$ and $y_1 < y_2$. Recall that parameters c and d are determined by equations (5.4) and (5.5) as

$$\begin{aligned} c &= p_1 + q_1 \times e; \\ d &= p_2 + q_2 \times e, \end{aligned}$$

where e is a bifurcation parameter and p_1, p_2, q_1 and q_2 are randomly chosen constants.

Since the chaotic map must be bounded, careful selection of the parameter r must be done to ensure the phase space is in a closed interval. Rewrite Equation (5.12) as

$$x_{n+1} = rh(x_n)(1 - h(x_n)) = rh(x_n) - r[h(x_n)]^2, \tag{5.13}$$

where $h(x_n) = \left(\frac{x_n - y_1}{y_m - y_1}\right)^c \left(\frac{y_2 - x_n}{y_2 - y_m}\right)^d$. To obtain the maximum value of x_{n+1}, solve x_n by computing

$$\frac{dx_{n+1}}{dx_n} = rh'(x_n) - 2rh(x_n) \cdot h'(x_n) = rh'(x_n)[1 - 2h(x_n)] = 0, \tag{5.14}$$

where

$$\begin{aligned} h'(x_n) = \quad & d\left(\frac{x_n - y_1}{y_m - y_1}\right)^c \left(\frac{y_2 - x_n}{y_2 - y_m}\right)^{d-1} \left(-\frac{1}{y_2 - y_m}\right) + \\ & c\left(\frac{y_2 - x_n}{y_2 - y_m}\right)^d \left(\frac{x_n - y_1}{y_m - y_1}\right)^{c-1} \left(\frac{1}{y_m - y_1}\right) \\ = \quad & \frac{(x_n - y_1)^{c-1}(y_2 - x_n)^{d-1}}{(y_m - y_1)^c (y_2 - y_m)^d} \cdot [c(y_2 - x_n) - d(x_n - y_1)]. \end{aligned} \tag{5.15}$$

When $r = 0$, $x_{n+1} = 0$ regardless the value of x_n.

Next, when $h'(x_n) = 0$, we obtain

$$x_n = \frac{cy_2 + dy_1}{c+d}. \tag{5.16}$$

This obtained x_n equals to y_m in Equation (5.12). Then, we substitute y_m into Equation (5.12) and obtain

$$r\left(\frac{y_m - y_1}{y_m - y_1}\right)^c \left(\frac{y_2 - y_m}{y_2 - y_m}\right)^d \left[1 - \left(\frac{y_m - y_1}{y_m - y_1}\right)^c \left(\frac{y_2 - y_m}{y_2 - y_m}\right)^d\right] = 0. \tag{5.17}$$

Therefore, when $x_n = y_m$, we will get $x_{n+1} = 0$ regardless the value of r. We could not determine the range of r for this case.

So, we look at the final case, i.e. when $1 - 2h(x_n) = 0$. We have

$$\begin{aligned}
\left(\tfrac{x_n - y_1}{y_m - y_1}\right)^c \left(\tfrac{y_2 - x_n}{y_2 - y_m}\right)^d &= \frac{1}{2} \\
(x_n - y_1)^c (y_2 - x_n)^d &= \frac{1}{2}(y_m - y_1)^c (y_2 - y_m)^d \\
&= \big(\frac{1}{2^{1/2c}}(y_m - y_1)\big)^c \big(\frac{1}{2^{1/2d}}(y_2 - y_m)\big)^d \\
&= \left[\left(\frac{1}{2^{1/2c}} y_m + \big(1 - \frac{1}{2^{1/2c}}\big) y_1\right) - y_1\right]^c \times \\
&\quad \left[y_2 - \left(\big(1 - \frac{1}{2^{1/2d}}\big) y_2 + \frac{1}{2^{1/2d}} y_m\right)\right]^d.
\end{aligned} \tag{5.18}$$

So,

$$x_n = \frac{1}{2^{1/2c}} y_m + \big(1 - \frac{1}{2^{1/2c}}\big) y_1 = \big(1 - \frac{1}{2^{1/2d}}\big) y_2 + \frac{1}{2^{1/2d}} y_m. \tag{5.19}$$

To make sure $x_{n+1} \in [0,1]$, we determine r by substituting Equation (5.19) into Equation (5.12). Let $x_{n,1} = \frac{1}{2^{1/2c}} y_m + \big(1 - \frac{1}{2^{1/2c}}\big) y_1$ and $x_{n,2} = \big(1 - \frac{1}{2^{1/2d}}\big) y_2 +$

$\frac{1}{2^{1/2d}}y_m$. Then, we obtain the range of r as

$$\begin{aligned}
0 \leq r\left(\frac{x_{n,1}-y_1}{y_m-y_1}\right)^c \left(\frac{y_2-x_{n,2}}{y_2-y_m}\right)^d \left[1-\left(\frac{x_{n,1}-y_1}{y_m-y_1}\right)^c \left(\frac{y_2-x_{n,2}}{y_2-y_m}\right)^d\right] &\leq 1 \\
0 \leq r\left(\frac{\frac{1}{2^{1/2c}}(y_m-y_1)}{y_m-y_1}\right)^c \left(\frac{\frac{1}{2^{1/2d}}(y_2-y_m)}{y_2-y_m}\right)^d \left[1-\left(\frac{\frac{1}{2^{1/2c}}(y_m-y_1)}{y_m-y_1}\right)^c \left(\frac{\frac{1}{2^{1/2d}}(y_2-y_m)}{y_2-y_m}\right)^d\right] &\leq 1 \\
0 \leq r(\frac{1}{\sqrt{2}})(\frac{1}{\sqrt{2}})(1-(\frac{1}{\sqrt{2}})(\frac{1}{\sqrt{2}})) &\leq 1 \\
0 \leq r &\leq 4.
\end{aligned} \tag{5.20}$$

From Equation (5.20), we have $r \in [0,4]$ which is same as the Logistic map.

5.4 Dynamical performance

In this section, we will characterize the dynamics of Logistic-Beta map geometrically with the trajectory and bifurcation plots, and statistically with the Lyapunov exponent.

5.4.1 Trajectory

Trajectory or orbit presents the moving path of the set of all points in the dynamical system (Kocarev and Lian, 2011). We show the trajectories for the Logistic-Beta map and the chaotic maps discussed in Section 5.2.

For Logistic-Beta and Beta maps, we set the initial values, $x_0 = 0$, as shown in Figures 5.1a and 5.1c. While the initial values for Logistic map and modified Logistic maps, $x_0 = 0.1$ and their trajectories are plotted in Figures 5.1b and 5.1d. As shown in Figure 5.1a, Logistic-Beta map has a larger distribution area as compared to Logistic and Beta maps, referring to Figures 5.1b and 5.1c. Even though the modified Logistic map in Figure 5.1d has a wider range for x_{n+1}, the outputs are not random and lack of dispersion. Therefore, Logistic-Beta map produces more random output and demonstrates a better ergodicity.

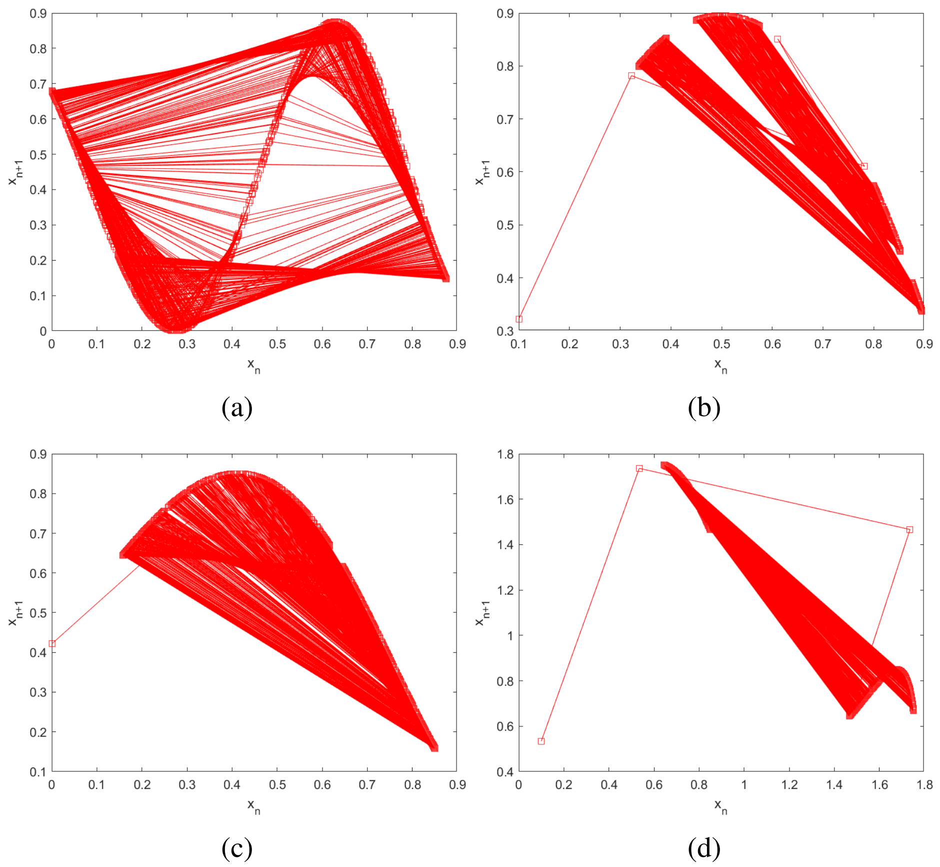
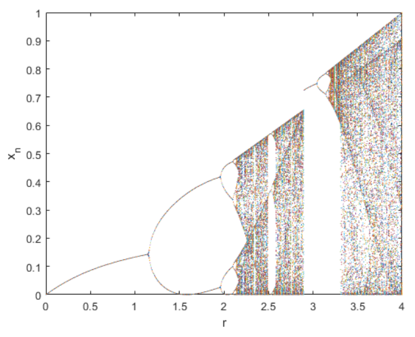


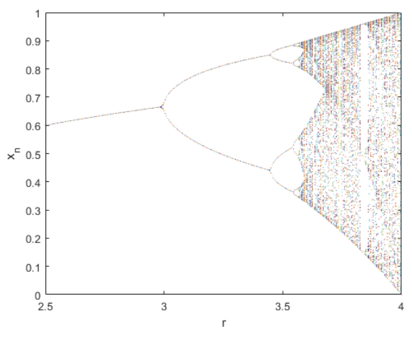
Figure 5.1: Trajectory Diagram: **(a)** Logistic-Beta map with $r = 3.5, e = 0.1, y_1 = -1, y_2 = 1, p_1 = 5, p_2 = 3, q_1 = 1, q_2 = -1$ **(b)** Logistic map with $r = 3.58$ **(c)** Beta-map with $\mu = 0.85, e = 0.65, y_1 = -1, y_2 = 1, p_1 = 5, p_2 = 3, q_1 = 1, q_2 = -1$ **(d)** Modified Logistic map with $r = 2$

5.4.2 Bifurcation diagram

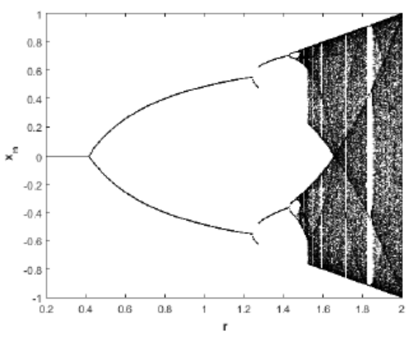
Bifurcation shows a qualitative change in dynamics for the variation of the control parameters of a dynamical system (Kocarev and Lian, 2011). In other word, the dotted area of the diagram describes the chaotic behavior of the system. As shown in Equation (5.12), Logistic-Beta map consists of two control parameters, i.e. r from Logistic map in Equation (5.1) and e from Beta map in Equation (5.4) and (5.5). We first vary the parameter r and shows the bifurcation diagram for $r = [0,4]$ in Figure 5.2a. When the parameters exceed the critical value, i.e. $r = 1.155$, the Logistic-Beta map exhibits a period-doubling bifurcation by converting the attractor from a period-1 firing to a period-2 firing. The following period-doubling bifurcations occur at $r = 1.95$, 2.1, 2.53, and 3.04. The dotted area in between bifurcations shows that the onset of chaos as various curves start merging together. As shown in Figure 5.2,



(a)



(b)

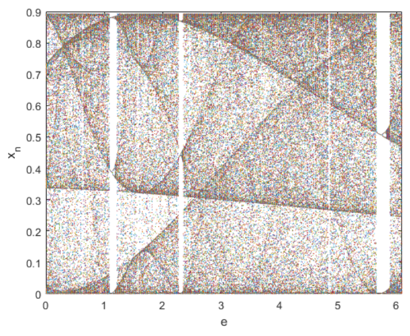


(c)

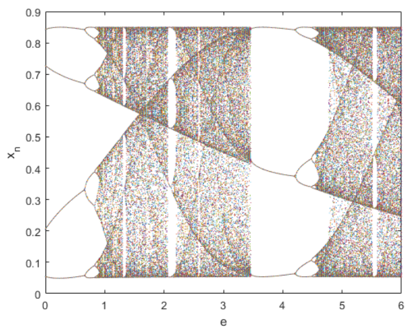
Figure 5.2: Bifurcation diagram of chaotic maps with different bifurcation parameters: **(a)** Logistics-Beta map with $0 \leq r \leq 4, e = 0.4, y_1 = -1, y_2 = 1, p_1 = 4, p_2 = 2, q_1 = 1, q_2 = 0.2$ **(b)** Logistics map with $2.5 \leq r \leq 4$ **(c)** Modified Logistic map with $0.2 \leq r \leq 2$

the logistic-based chaotic maps consists of windows of periodic behaviors causing the maps vulnerable to parameter estimation attacks (Arroyo et al., 2010).

Logistic-Beta map has an advantage over the other maps as it has another control parameter e which enlarges the phase space and make the proposed map more chaotic. We compare the bifurcation diagram of Beta map and Logistic-Beta map by varying parameters e, refer to Figures 5.3 for the comparison. As shown in Figure 5.3a, the proposed map has excellent chaotic behavior along the range $e \in [0,6]$ as it has a very few periodic windows as compared to Beta and the dotted points are scattered around the area.



(a)



(b)

Figure 5.3: Bifurcation diagram of chaotic maps with different bifurcation parameters: **(a)** Logistics-Beta map with $0 \leq e \leq 9$ and $r = 3.57, y_1 = -1, y_2 = 1, p_1 = 4, p_2 = 2, q_1 = 1, q_2 = 0.2$ **(b)** Beta map with $0 \leq e \leq 9, \mu = 0.85, y_1 = -1, y_2 = 1, p_1 = 4, p_2 = 2, q_1 = 1, q_2 = 0.2$

5.4.3 Lyapunov exponent

Lyapunov exponent (LE) is a quantitative measure to test the sensitivity of the chaotic map to the slight changes in the initial conditions and control parameters (Kocarev and Lian, 2011). A positive LE indicates that the chaotic map has a good chaotic behavior, and the higher the LE value shows a better sensitivity of the map to its initial value or system parameters. From Figure 5.4a, it is obvious that Logistic-Beta map has the highest LE value and also a greater chaotic range, i.e. it has positive LE for $e > 2.3$. As shown in Figures 5.4b and 5.4c, Logistic map has positive LE for $r \in [3.57,4]$ while Beta map has positive LE values for

$e > 3.2$. For modified Logistic map in Figure 5.4d, the chaotic map only have positive LE when $r \in [-2, -1.5] \cup [1.5, 2]$.

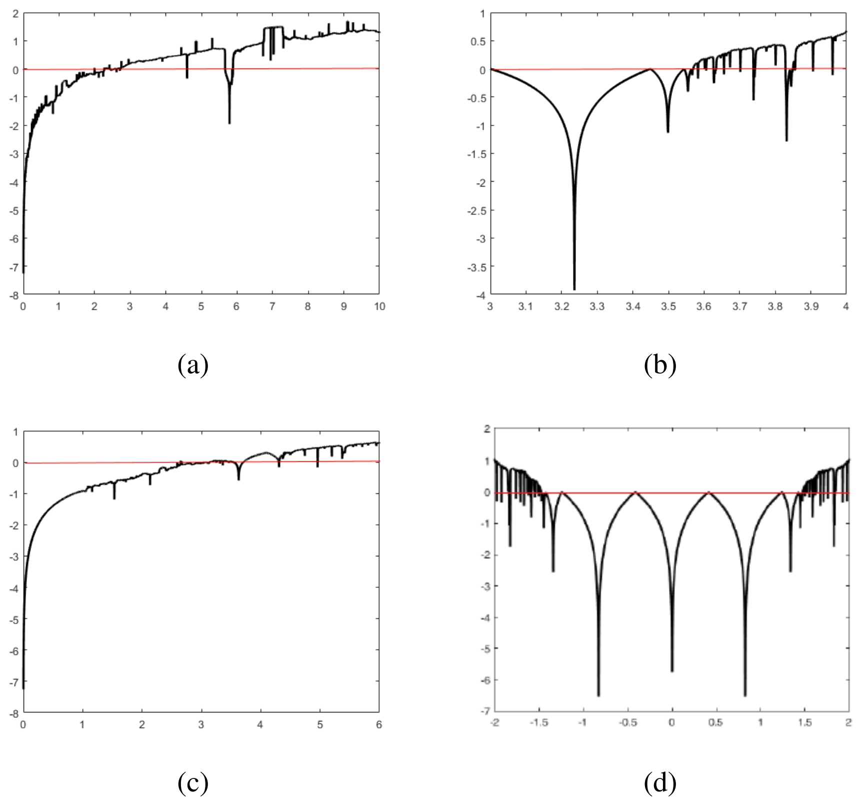


Figure 5.4: Lyapunov Exponent: **(a)** Logistic-Beta map with $e \in [0, 10], r = 3.57, y_1 = -1, y_2 = 1, p_1 = 4, p_2 = 2, q_1 = 1, q_2 = 0.2$, **(b)** Logistic map with $r \in [3, 4]$, **(c)** Beta-map with $e \in [0, 6], \mu = 0.85, y_1 = -1, y_2 = 1, p_1 = 4, p_2 = 2, q_1 = 1, q_2 = 0.2$, **(d)** Modified Logistic map with $r = \pm 2$

5.5 Summary

This chapter proposes a new chaotic map, called Logistic-Beta map. We have proven that the proposed chaotic map has significantly improved the chaotic behaviors of classical Logistic and Beta maps. The chaotic behaviors of proposed chaotic map also have been discussed and compared with modified Logistic map which is a chaotic map designed based on Logistic map. The advantages of Logistic-Beta map is summarized as follows.

1. The large distribution area in the phase plane shows that our map has a good ergodicity.
2. The large darked area in the bifurcation diagram demonstrates that the proposed map has a large chaotic region, leading to a large key space.
3. A positive Lyapunov Exponent value indicates that our proposed map has good sensitivity to initial values.

These advantages make the proposed chaotic map suitable to be applied in an image encryption scheme.

CHAPTER 6

DYNAMICS ANALYSIS OF A TWO-DIMENSIONAL CHAOTIC MAP

This chapter focuses on the design and application of a two-dimensional chaotic map that demonstrates enhanced chaotic behavior. It introduces a new chaotic map developed through cascading methods to overcome the limitations identified in previous chapters. The goal is to strengthen the chaotic properties of the map for cryptographic applications.

Building on previous findings, this work is driven by two key observations. First, Chapter 4 analyzed Ping et al.'s scheme, which uses the Henon map to generate pseudorandom sequences. The Henon map serves as the foundation for both the permutation and diffusion processes in this scheme. Under specific parameter settings and finite precision environments, its chaotic behavior deteriorates, resulting in reduced unpredictability and randomness. This degradation weakens its security, making it vulnerable to cryptographic attacks. Second, Chapter 5 revealed periodic windows in one-dimensional chaotic maps, including the Logistic and Beta maps. Even the proposed Logistic-Beta map exhibits some periodic windows, compromising its effectiveness for encryption. These weaknesses emphasize the need for more robust chaotic systems.

To address these issues, this chapter proposes a novel two-dimensional chaotic map by cascading chaotic maps with modular components, enhancing their overall dynamical properties. The resulting two-dimensional improved modular chaotic map (2D-IMCM) shows greater sensitivity to initial conditions, enhanced randomness, and a wider chaotic range than traditional maps. These improved features are utilized to generate high-quality pseudorandom number sequences that are suitable for cryptographic applications. Furthermore, this work lays the foundation for Chapter 8, where a novel two-dimensional

Sine-Henon map is proposed and applied to a color image encryption scheme.

6.1 Introduction

To solve the dynamical degradation problem, many chaotification approaches have been proposed by researchers, for example, perturbing chaotic states or parameters (Luo et al., 2021; Liu et al., 2020), linear feedback control (Liu et al., 2020; Liu and Liu, 2020), coupling (Liu and Liu, 2020; Pak et al., 2021), and cascading of multiple chaotic maps (Pak et al., 2021; Zhang, Ding and Li, 2020; Wong, Yap, Goi and Wong, 2020). Comparison of these approaches is summarized in Table 6.1.

Table 6.1: Comparisons of Chaotification Approaches

Chaotification Methods	**Characteristics**	**Limitations**
Perturbation	Use external perturbation sources	High computational cost
Feedback control	Use state function to manipulate the trajectory of chaotic map	Have to work with other methods to improve the chaoticity
Coupling	Combine chaotic maps	High computational cost
Cascading	Use outputs of one chaotic map as the state variable of another map	Regular dynamics

Comparing to other approaches that require external sources or high computational costs, cascading technique combines two or more chaotic maps which can significantly improve the complexity of dynamics characteristics with more flexibility and prolong the cycle length. However, the cascading chaotic systems proposed in (Pak et al., 2021; Zhang, Ding and Li, 2020; Wong, Yap, Goi and Wong, 2020) that used one-dimensional chaotic maps as the seed map did not have complex structures. Using Henon map as the seed map in (Wu et al., 2021) also did not give a good result as the chaotic range of the control parameters is not broad enough.

The main novelty of this chapter is as follows: (1) We demonstrate the dynamical degradation of the Henon map in the fixed-point arithmetic domain using state-mapping network. (2) We apply cascading approach to construct a new 2-dimensional (2D) chaotic system to produce the pseudorandom sequences without the usage of external sources and low computational cost. (3) A pseudorandom number generating algorithm is designed. The output passed all the subtests in the NIST SP800-22 indicating that it has a good randomness.
Organization: The remainder of this chapter is organised as follows. Section 6.2 describes the preliminaries of some exisiting chaotic maps. Section 6.3 discusses the dynamical degradation of Henon map. Section 6.4 presents the mathematical model of our newly proposed chaotic map and dynamical analysis of our proposed map. Section 6.5 discusses the application of our proposed map to pseudorandom number generator. The last section summarizes the chapter.

6.2 Preliminaries

In this section, we first present the equations of the existing chaotic maps which will be applied in designing the new chaotic map. A 2D chaotic map, Henon map will be used as the seed map, while the logistic map will be used as one of the state variables of 2D chaotic map. We also introduce an 2D modular chaotification system (2D-MCS) proposed by (Hua et al., 2020) for the performance comparison purpose. Given x and y are two state variables, and $n \in \mathbf{Z}^+$ is the number of iterations.

Henon map

Recall from Equation (4.1) that the Henon map is mathematically represented by

$$\begin{cases} x_{d+1} = 1 - ax_d^2 + y_d; \\ y_{d+1} = bx_d, \end{cases}$$

where $a = 1.4$ and $b = 0.3$ are the control parameters and d is the d-th iteration of the chaotic map. Lastly, x_0 and y_0 are the initial values of the chaotic map.

Logistic map

Recall from Equation (5.1), Logistic map is represented by

$$x_{d+1} = \mu x_d(1 - x_d), \tag{6.1}$$

where μ is the control parameter.

2D-MCS

2D-MCS is proposed by Hua et al. (2020) and it is represented by

$$M(x,y) = F(x,y) \bmod N, \tag{6.2}$$

where $F(x,y)$ is a 2D chaotic map and N is a positive integer. We let $F(x,y)$ be the Henon map where they are given in Equations (4.1). The resulting chaotic maps are named as Improved Henon map.

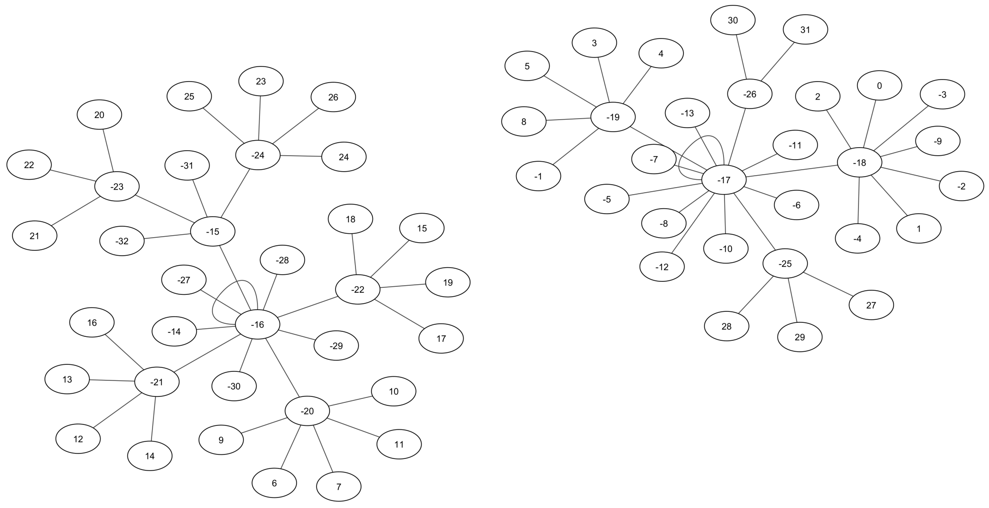
6.3 Dynamical degradation of Henon map

As mentioned in Subsection 4.2.1 of Chapter 4, the Henon map from Equation (4.1) is utilized to generate pseudorandom sequences for keystream generation. However, when the Henon map is implemented on digital computers with limited precision, the dynamic properties of the continuous chaotic map may not be preserved. This section further explores the dynamical properties of the Henon map through the state-mapping network (SMN).

According to Li, Feng, Li, Kurths and Chen (2019) and Li et al. (2021), using the fixed point precision of n and quantifization, the SMN is built with $(2^n)^2$ possible states. For illustration, SMNs of Henon map are plotted with the control parameters $a = 1.4, b = 0.3$ and the precision of $n = 3, 4$ using Matlab R2019a environment. For $n = 3$ or 6-bit precision, the output of each iteration of the henon map will fall into one of the $(2^3)^2 = 64$ states. For $n = 4$ or 8-bit precision, it will have $(2^4)^2 = 256$ possible states. As shown in Figure 6.1, the characteristics of the iteration trajectories of the Henon map are listed below:

- All the initial states numbered from $[-2^n, 2^n - 1]$ converge to fixed points through the transient process. The fixed points are referring to self-loops in the SMN, i.e. -16 and -17 for $n = 3$, and -47 and -49 for $n = 4$.
- The transient length, namely the distrance from a leaf nodes to a root node is very short. Many initial states converge to the fixed point in only one or two iterations.

Based on the observations, different initial values of the henon map can produce the same chaotic sequence after a short iteration. This indicates that numerous invalid and equivalent keys may lead to identical chaotic sequences. As a result, the key space is significantly smaller than anticipated in practical applications. To address this issue, one might consider utilizing a chaotic system with greater


(a) $n = 3$

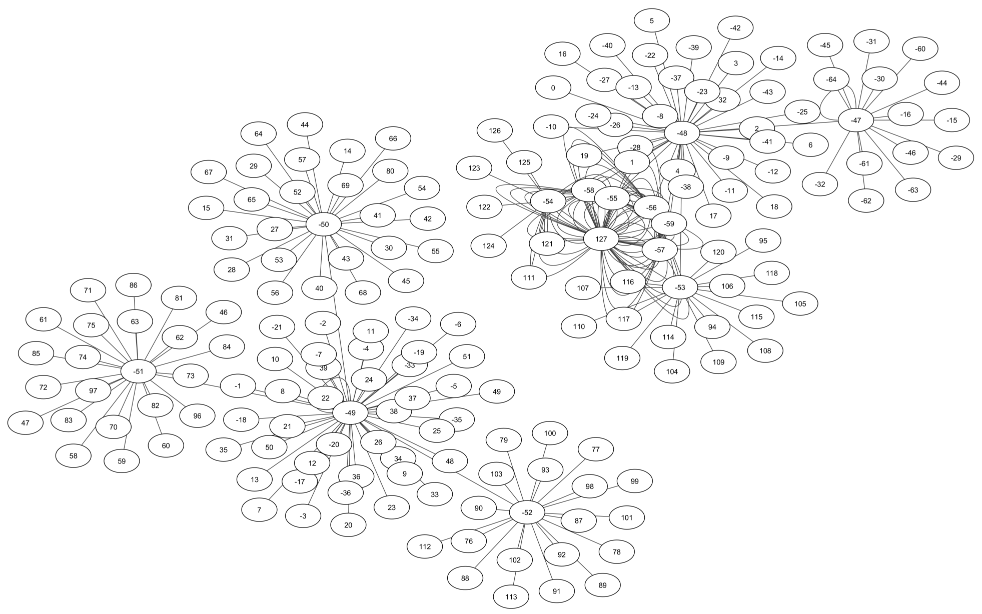

(b) $n = 4$

Figure 6.1: State-mapping networks of Henon map with $a = 1.4$ and $b = 0.3$, implemented under different fixed-point precisions n

complexity and a longer cycle length, which can withstand dynamic degradation in real-world scenarios.

6.4 Two-dimensional improved modular chaotic map

The 2-dimensional improved modular chaotic map (2D-IMCM) proposed in this chapter is expressed by

$$\begin{cases} x_{1,d+1} = g(x_{1,d}, f(x_{2,d} + k) \bmod N; \\ x_{2,d+1} = g(x_{1,d}, f(x_{2,d} + k) \bmod N, \end{cases} \tag{6.3}$$

Under this proposal, a 2-dimensional chaotic system is cascaded with other function to prolong the cycle length. In other words, the output of f is used to initiate the pseudo trajectory of another function, g for every iteration. So, $f(\cdot)$ denotes the logistic map while $g(\cdot)$ denote the seed map which is the 2D chaotic map.

To show the effectiveness of 2D-IMCM, we use Henon map as the seed map. We name the resulting chaotic map as 2D-Henonlog map. The equation of the proposed 2D-Henonlog map is given as

$$\begin{cases} x_{d+1} = 1 - ax_d^2 + \mu y_d(1 - y_d) + k \bmod N; \\ y_{d+1} = bx_d \bmod N, \end{cases} \tag{6.4}$$

where μ, a and b are control parameters, N is positive integer, and k is a constant to improve the complexity.

6.4.1 Proof of chaoticity

To prove a dynamical system is chaotic, we must fulfill two criteria: (1) The chaotic map must be bounded. (2) There must have at least one positive Lyapunov Exponent (LE). If a dynamical system has at least one positive LE, then the two extremely close trajectories will diverge in multi-direction over the time, and make the dynamical system become chaotic.

The first criterion is fulfilled because the equation involves modular arithmetic, so the outputs are bounded as $0 \leq x_d, y_d \leq N$, where N is the number of iterations. To show 2D-IMCM fulfills the second criterion, we use the method proposed by Alawida et al. (2019). Let g be the function of 2D chaotic map and f be the function of logistic map. The LE of 2D-IMCM is given by

$$\begin{aligned} LE &= \lim_{n\to\infty} \frac{1}{n} \sum_{d=0}^{n-1} \ln \left| \frac{dg}{dx} |_{f(x_d)} \times \frac{df}{dx} |_{x_d} \right| \\ &= \lim_{n\to\infty} \left(\ln \left| \frac{dg}{dx} |_{f(x_d)} + \ln | \frac{df}{dx} |_{x_d} \right| \right) \\ &= \lim_{n\to\infty} \left(\ln \left| \frac{dg}{dx} |_{f(x_d)} \right| \right) + \lim_{n\to\infty} \left(\ln \left| \frac{df}{dx} |_{x_d} \right| \right). \end{aligned} \tag{6.5}$$

When both LE_g and LE_f are greater than zero, then LE of 2D-IMCM will be greater than zero, meaning that 2D-IMCM is chaotic. To make LE_g, $LE_f > 0$, the selection of control parameters within the chaotic region of the chaotic map is important. Henon map is chaotic when $a = 1.4$ and $b = 0.3$, whereas the chaotic region of logistic map is $\mu \in (3.5699456, 4)$. As shown in Figures 6.2a and 6.2b, both chaotic maps have one positive LE.

LE is also an indicator to show the sensitivity to the initial conditions. The higher the LE values of the chaotic maps, the better its sensitivity. From Figures 6.2a and 6.2c, it is observed that the LE value of 2D-Henonlog map is 1.7119 which is larger than LE value of Henon map, 0.42311. Thus, our proposed 2D-

IMCM has better chaotic behavior than its underlying map.

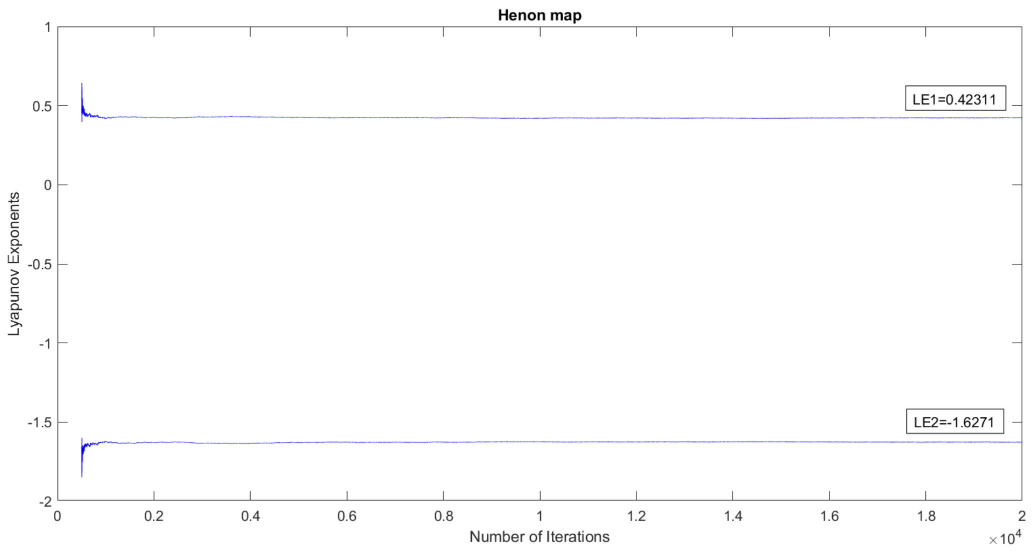
6.4.2 Performance evaluations

In this section, we present the dynamics of 2D-IMCM geometrically using the bifurcation plots to visualize the chaotic range of the 2D-IMCM. We also plot the state mapping network of the 2D-IMCM to study its periodicity. In the study, we compare their performance with the existing 2D chaotic system, i.e., Henon map and Improved Henon map proposed by Hua et al. (2020).

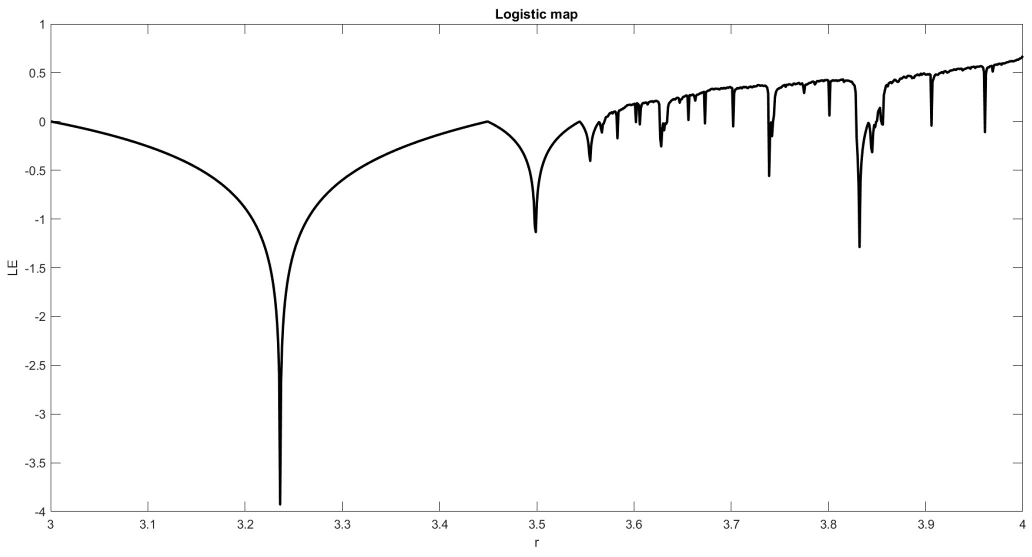
Bifurcation Diagram

Bifurcation diagram is a test showing the dynamical change of a chaotic map when the control parameters change (Kocarev and Lian, 2011). The dotted area scattered on the diagram indicates the chaotic area of a chaotic map. In the experiment, we set $x_0 = 0.1, y_0 = 0.1, a = 1.4, b = 0.3, k = 3, \mu = 3.67$ for Henon, Improved Henon and 2D-Henonlog maps. Figures 6.3a-6.3c shows the bifurcation diagrams by changing parameter a, Figures 6.3d-6.3f demonstrates the bifurcation diagrams by changing parameter b. It can be seen that Henon map in 6.3a & 6.3d and Improved Henon map in Figure 6.3b & 6.3e exist multiple periodic windows which shows non-chaotic regions on the discontinuous chaotic ranges.

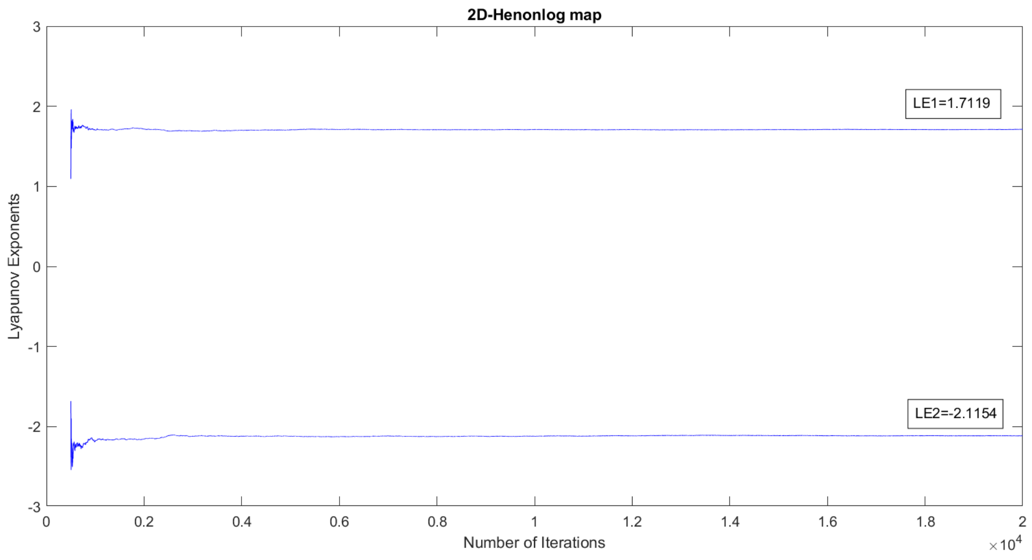
From Figure 6.3c and 6.3f, it is observed that the outputs of our proposed 2D-Henonlog map are randomly distributed on the entire phase plane for a wide range of parameters. Besides, 2D-Henonlog also has another advantage of having additional parameters μ and k which can widen the phase space and make the dynamical system more chaotic. We vary the parameter μ and k, and



(a)



(b)



(c)

Figure 6.2: Lyapunov Exponents of underlying seed maps

show the bifurcation diagrams in Figures 6.3g and 6.3h. Apparently, 2D-Henonlog has excellent chaotic behaviors because the dotted points are scattered evenly in the whole range of the parameters.

State mapping network

To investigate the dynamical degradation of the chaotic map, we use the state mapping network (SMN) to observe iterative trajectories of the digital chaotic maps. The periodic distribution of chaotic maps is studied in the following aspects: the maximal transient length, cycle length, number of cycles, and the number of fixed points. In the study, the SMNs are drawn with $(2^n)^2$ possible states, where n is the finite computational precision. We use $n = 3$ in the experiment, meaning that the outputs of the chaotic maps will fall into 64 states. We also choose the parameters where the chaotic maps are chaotic, refer to Figures 6.1a, 6.4a and 6.4b for SMNs of Henon map, Improved Henon map and 2D-Henonlog map, respectively. The results of periodic distribution are compared and summarized in Table 6.2.

We observe that Henon map and 2D-Henonlog map has two cycles, whereas Improved Henon map has three cycles. The maximal transient length measures the largest number of iterations of the state variables before entering into a cycle. From Figures 6.4a and 6.4b, it is observed that 2D-Henonlog map has the same maximal transient length with the Improved Henon map. Both chaotic maps enter into a periodic cycle after 8 iteration operations, but the cycle length of 2D-Henonlog map is longer. Besides, 2D-Henonlog map also does not have fixed point. The fixed point is represented by the self-loops in the SMNs. From Figures 6.1a and 6.4a, we can see that Henon and Improved Henon map tends to converge to the fixed points after limited number of iterations. This means that there might exist equivalent keys that would result in same chaotic sequences. The results show that our proposed chaotic maps are outperformed than the other

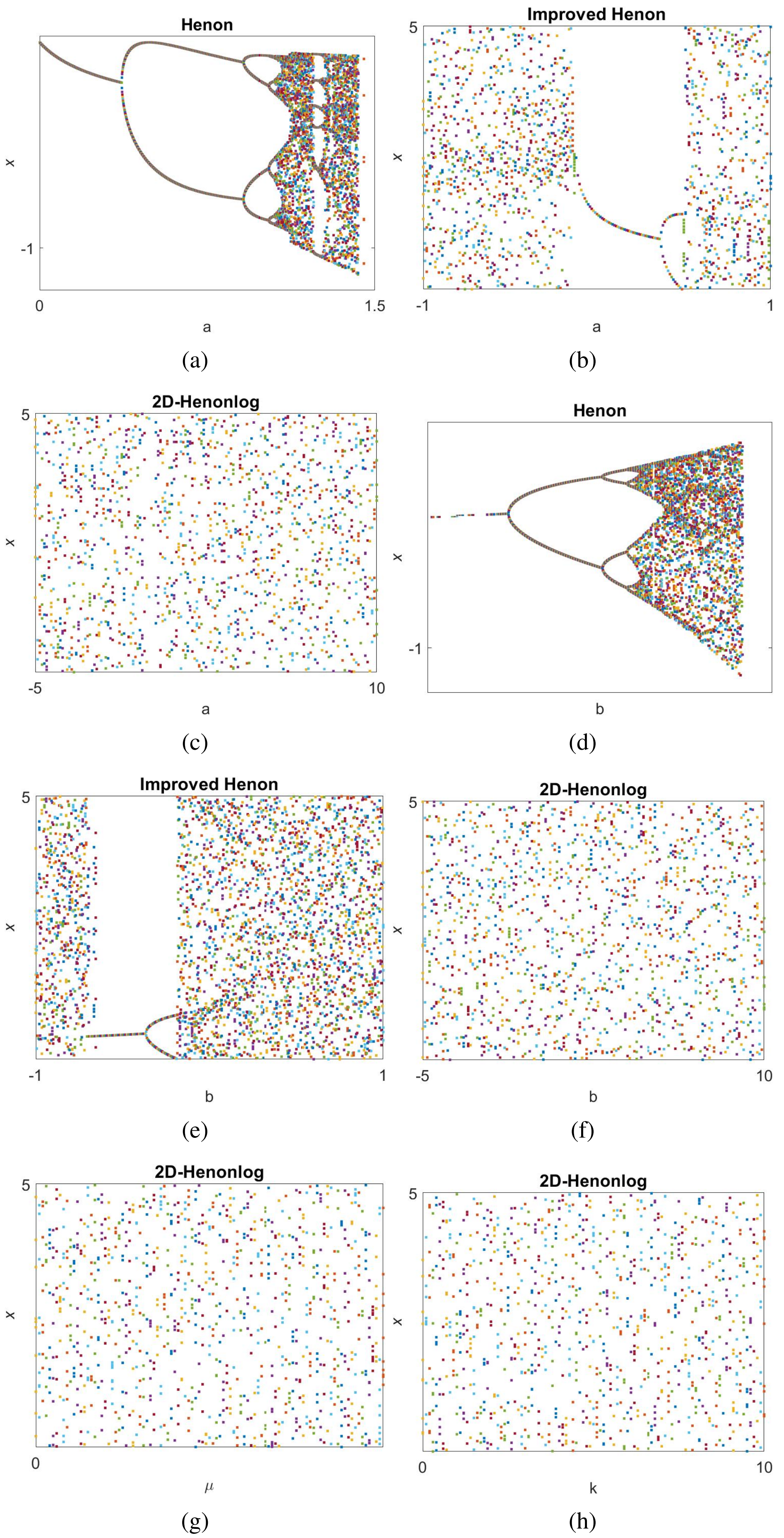
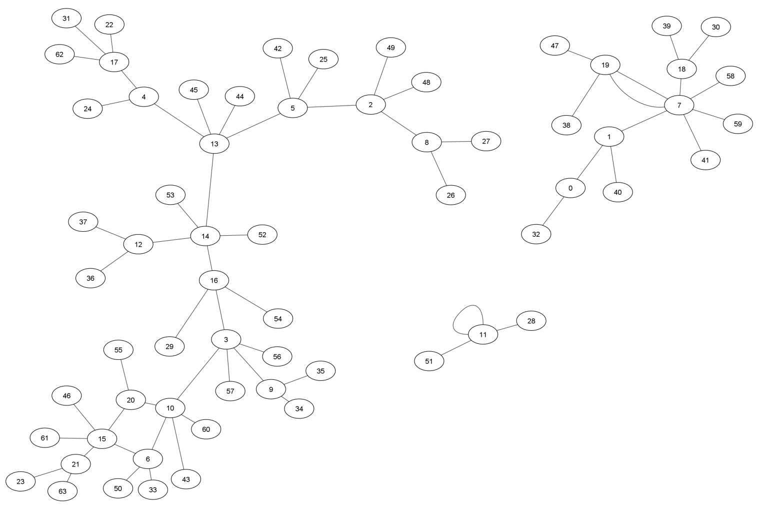


Figure 6.3: Bifurcation diagram of chaotic maps

the chaotic maps.


(a) State-mapping network of Improved Henon map

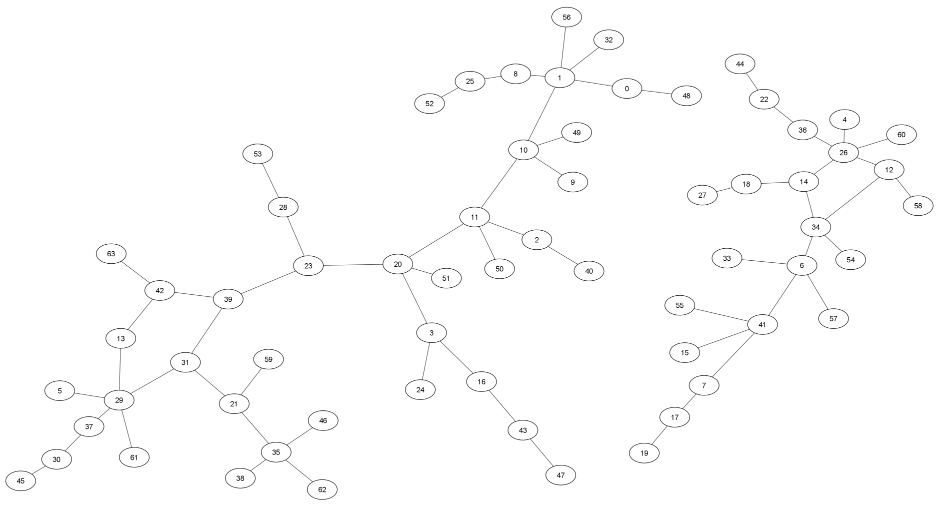

(b) State-mapping network of 2D-Henonlog map

Figure 6.4: SMN of chaotic maps

6.5 Application to pseudorandom number generator

Chaotic map is commonly used to design a pseudorandom number generator (PRNG) whereby the sequence produced by PRNG is useful to build the permutation vector and perform encryption operations of the image encryption

Table 6.2: Comparisons of period distribution

Chaotic map (control parameters)	Max. transient length	Max. cycle length	No. of cycle	No. of fixed points
(a) Henon	3	0	2	2
(b) Improved Henon	8	4	3	1
(c) 2D-HenonLog	8	5	2	0

algorithm. This section presents a PRNG designing algorithm based on 2D-Henonlog. The algorithm is given as follows.

1. Set the initial conditions $(x_0 = 0, y_0 = 0)$ and control parameters of the 2D-Henonlog $(a = 1.4, b = 0.7, k = 3, \mu = 3.67)$ given in Equation (6.4)).
2. Iterate the chaotic maps for 10^6 times.
3. Apply the following formula to the outputs of 2D-Henonlog.

$$\mathbf{X} = \{\text{floor}(x_d \times 10^{15})(\bmod 256)\}_{d=1}^{10^6},$$
$$\mathbf{Y} = \{\text{floor}(y_d \times 10^{15})(\bmod 256)\}_{d=1}^{10^6}.$$

4. Obtain sequences $\mathbf{X}'$ and $\mathbf{Y}'$ by converting the sequences $\mathbf{X}$ and $\mathbf{Y}$ to an 8-bit binary array.

To test the pseudorandom-like behavior of the PRNG produced from the algorithm above, we perform the randomness test by using the National Standard and Technology Institute (NIST) SP800-22 test suite (Bassham III et al., 2010). It consists of 15 subtests with passing criteria that the p-value of each test must be greater than or equal to the significance level of α. In our experiment, we set $\alpha = 0.01$ and 10^6 bit in the bitstream length. The results are presented in Table 6.3. It shows that the PRNG produced by our proposed chaotic maps passed all the subtests.

[1]denotes the average values of the respective tests.

Table 6.3: NIST SP800-22 Test Results

Test	p-value
Frequency	0.122325
Block Frequency	0.213309
Cumulative Sums[1]	0.350485
Runs Test	0.534146
Longest Run	0.350485
Binary Matrix Rank	0.739918
FFT	0.534146
Non-overlapping Template[1]	0.494351
Overlapping Template	0.534146
Universal	0.534146
Approximate Entropy	0.122325
Random Excursions[1]	0.482017
Random Excursions Variant[1]	0.232147
Serial[1]	0.630949
Linear Complexity	0.534146

6.6 Summary

This chapter presents a new two-dimensional chaotic map based on cascading technique and modular operation. Henon map was chosen as the examples to show the effectiveness of the proposed chaotic map. The chaoticity of the chaotic map was proven. The dynamical performance was shown by using bifurcation diagram and state mapping network. The results show that our proposed chaotic map possesses a better chaotic behavior as compared to the underlying seed maps and other existing chaotic maps. The chaotic map was applied to design a PRNG. The sequences produced by the PRNG passed all the subtests in the NIST SP800-22 test suite. Despite of that, there are still some tests can be carried on the produced sequences, e.g., complexity test, speed test, scale index, etc. Besides, we can further extend the research work to the application of the cascading chaotic map in the image encryption algorithm.

CHAPTER 7

A NEW IMAGE ENCRYPTION SCHEME BASED ON HYPERCHAOTIC SYSTEM AND SHA-2

This chapter builds upon the cryptanalysis conducted in Chapters 3 and 4 by introducing a new image encryption scheme that addresses the weaknesses identified in previous methods. Chapter 3 revealed that the genetic algorithm-based encryption scheme proposed by Biswas et al. has low sensitivity to changes in plaintext, which violates key cryptographic design principles such as nonlinearity. Additionally, Chapter 4 demonstrated that the scheme by Ping et al., which combines permutation and diffusion into a single process using the Henon map, suffers from dynamical degradation, making it vulnerable to chosen-plaintext attacks.

To address these limitations, this chapter presents an improved image encryption scheme that utilizes a hyperchaotic system and SHA-2. The hyperchaotic system increases unpredictability and expands the keyspace. Additionally, a nonlinear diffusion process is integrated to enhance security against differential attacks, which is a response to the weaknesses found in the sequential encryption methods of the targeted schemes. The incorporation of SHA-2 further elevates the sensitivity of the cipher to the plaintext, ensuring that even a slight change in the input results in a significantly different ciphertext. By implementing these improvements, the proposed encryption scheme establishes a more robust security framework and overcomes the vulnerabilities identified in prior schemes.

7.1 Introduction

The common chaotic based encryption scheme consists of two processes, i.e., permutation and diffusion to fulfill the confusion and diffusion properties (Shannon, 1949). The permutation-diffusion architecture that firstly proposed by Fridrich (1998) becomes the benchmark in image encryption and was widely utilized by many researchers. In the permutation process, the position of the image is changed at the pixel or bit levels and then the values are altered in the diffusion process. However, Fridrich's scheme with multi-round was cryptanalyzed by Solak et al. (2010) by chosen ciphertext attack and the cryptanalytic attack was further improved by Xie et al. (2017). Besides, there are many schemes found to be insecure (Boriga et al., 2014; Zhang et al., 2016; Biswas et al., 2015; Khan, 2015) and cryptanalyzed by chosen plaintext or ciphertext attacks (Wen et al., 2017; Wu et al., 2018a; Wong, Yap, Wong, Phan and Goi, 2020; Alanazi et al., 2021) due to the linear relationship between cipher and plain image and the independence of chaotic sequences from the plain image. Moreover, the widely applied diffusion mechanisms based on different combination of modular addition and exclusively-or operation have been discussed and cracked by Zhang et al. (2018) and Chen et al. (2021), respectively.

To overcome the weaknesses mentioned above, this chapter presents an image encryption based on a hyperchaotic system that modified from Lorenz chaotic attractor in keystream generation process (Zhang et al., 2017). The initial condition of the hyperchaotic system is generated by SHA-256 hash algorithm to avoid the chosen plaintext attack. A new nonlinear equation is used in the diffusion process to further enhance the security of the image encryption scheme. The remainder of this chapter is organized as follows. Firstly, the proposed image encryption scheme is discussed in details. Next, the simulation results and the security tests are discussed. Finally, the conclusion is drawn.

Organization: The remainder of this chapter is organised as follows. Section Section 7.2 presents a new image encryption scheme by using a four-dimensional hyperchaotic system and adopting permutation-diffusion architecture. Section 7.3 discusses the security analysis to show that the proposed scheme has large key and subkey space, high key sensitivity, good information entropy, and capability to resist statistical and differential attacks. The last section summarizes the chapter.

7.2 The proposed image encryption algorithm

A four-dimensional hyperchaotic system presented by Zhang et al. (2017) is applied in our scheme and the mathematical equation is given by

$$\begin{cases} \dot{x} = a(y-x) - ew, \\ \dot{y} = xz - hy, \\ \dot{z} = b - xy - cz, \\ \dot{w} = ky - dw, \end{cases} \tag{7.1}$$

where x, y, z, w are the state variable, and a, b, c, d, e, h and k are the control parameters.

The bifurcation diagrams of the hyperchaotic system given in Equation (7.1) are shown in Figure 7.1. The bifurcation measures the dynamical variation of the tiny change in one of the parameters. Fig. 7.1a–7.1g shows the bifurcation diagram with the change of parameter a, b, c, d, e, h and k, respectively. For example, Figure 7.1a shows the bifurcation diagram for the range of $0 \leq a \leq 25$, while the rest of the parameters remain unchanged. Besides of the changing parameter, the other parameters used in the plotting are $x_0 = 3.2$, $y_0 = 8.5$, $z_0 =$

3.5, $w_0 = 2.0$, $a = 5$, $b = 20$, $c = 1$, $d = 0.1$, $e = 20.6$, $h = 1$, $k = 0.1$, $x_0 = 3.2$, $y_0 = 8.5$, $z_0 = 3.5$ and $w_0 = 2.0$. The dotted points in the diagrams indicates the chaotic region of the system while the solid lines show the periodic region. The hyperchaotic system shows chaotic behaviors when $a > 0$, $b > 5$, $c \in (0,2)$, $d > 0$, $e > 0$, $h \in (0,4)$ and $k > 0$.

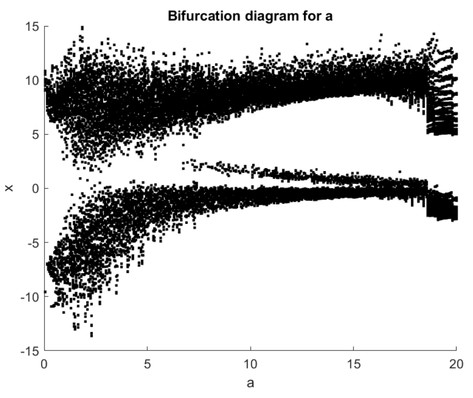
To increase the sensitivity of image encryption to plain image, SHA-256 hash algorithm (FIPS, 2001) is used to generate a 256-bit digest of the plain image, **K**. Slightly change in the plain image will results in a totally different digest. The digest is divided into 32 8-bit blocks as

$$\mathbf{K} = k_1, k_2, \ldots, k_{32}. \tag{7.2}$$

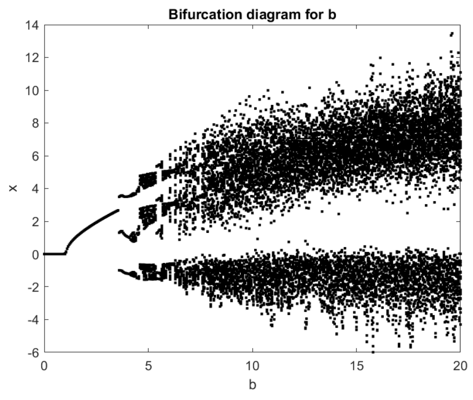
The initial conditions of the chaotic system, x_0, y_0, z_0 and w_0 are updated using **K** as

$$\begin{aligned} x_0' &= x_0 + \tfrac{(k_1 \oplus k_2 \oplus \cdots \oplus k_8)}{2^{16}}, \\ y_0' &= y_0 + \tfrac{(k_9 \oplus k_{10} \oplus \cdots \oplus k_{16})}{2^{16}}, \\ z_0' &= z_0 + \tfrac{(k_{17} \oplus k_{18} \oplus \cdots \oplus k_{24})}{2^{16}}, \\ w_0' &= w_0 + \tfrac{(k_{25} \oplus k_{26} \oplus \cdots \oplus k_{32})}{2^{16}}. \end{aligned} \tag{7.3}$$

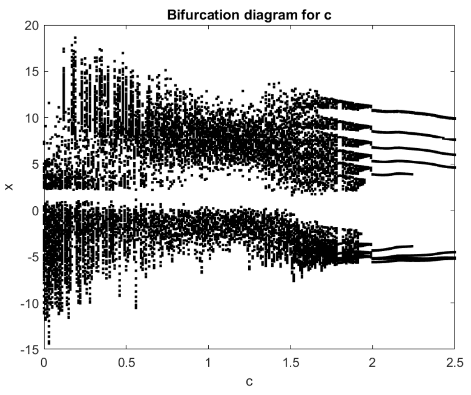
Then, the chaotic system is iterated for $N_0 + L$ times by using x_0', y_0', z_0', w_0', to generate four pseudorandom sequences. The first N_0 elements are removed to avoid harmful transient effect and four sequences $\mathbf{X} = \{x(i)\}_{i=0}^{L}$, $\mathbf{Y} = \{y(i)\}_{i=0}^{L}$, $\mathbf{Z} = \{z(i)\}_{i=0}^{L}$, and $\mathbf{W} = \{w(i)\}_{i=0}^{L}$ are obtained, where L is the size of the plain image. The sequences **X**, **Y**, **Z** and **W** will be used in permutation and diffusion stages which will be discussed in the subsections later.



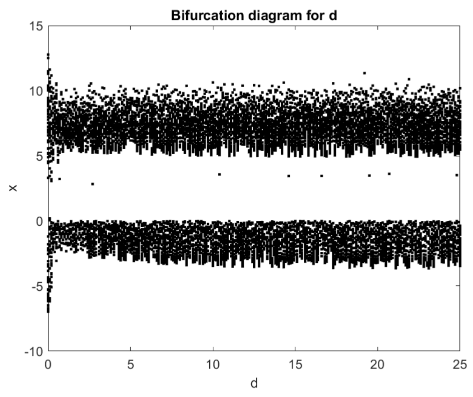
(a)



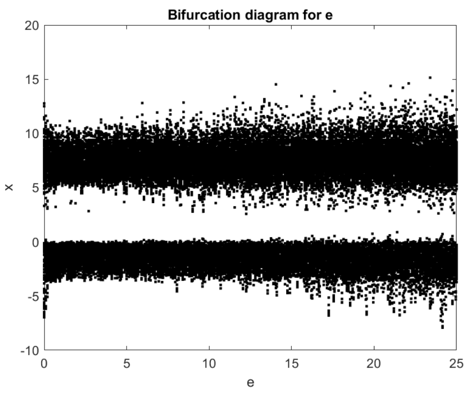
(b)



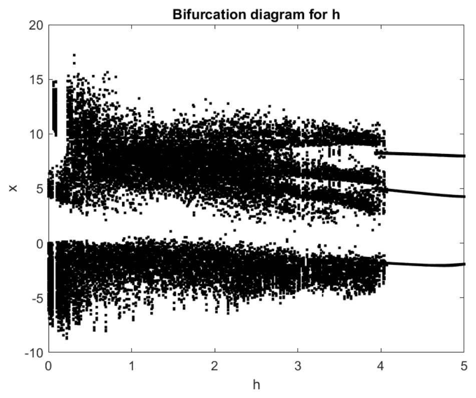
(c)



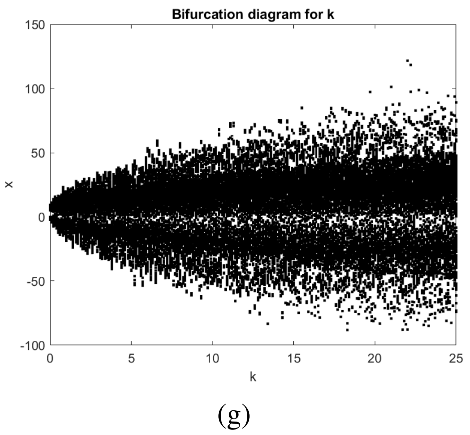
(d)



(e)



(f)



(g)

Figure 7.1: Bifurcation of hyperchaotic systems by changing the range for different control parameters

7.2.1 Permutation stage

Permutation is a process to change the position of the plain pixels in order to disrupt the correlation between the adjacent pixels of an image. Conventionally, the pixels are swapped by using two-dimensional area-preserving chaotic maps, for instance, baker's map and Arnold's cat map. However, the periodicity, existence of fixed points, and constrained cycle length of the chaotic maps jeopardize the efficiency and security level of the process.

To overcome this problem, the permutation vector is generated by the state variables from the hyperchaotic system given in Equation (7.1). Suppose the image pixels are scanned sequentially from left to right and from top to bottom rows. Without the loss of generality, the plain image with size, $L = M \times N$ is denoted as $\mathbf{P} = \{p(i)\}_{i=0}^{MN}$. The sequence $\mathbf{X} = \{x(i)\}_{i=0}^{MN}$, are changed to an integer sequence $\mathbf{X}' = \{x'(i)\}_{i=0}^{MN}$, where

$$x'(i) = \text{floor}([\text{abs}(x(i)) - \text{floor}(x(i))] \times 10^{15}). \tag{7.4}$$

The permutated image is produced by using Equations (7.5) and (7.6).

$$[\mathbf{V}, \mathbf{Idx}] = \text{sort}(\mathbf{X}'), \tag{7.5}$$

$$\mathbf{P}' = \mathbf{P}(\mathbf{Idx}). \tag{7.6}$$

where $\mathbf{V}$ is the new vector after sorting $\mathbf{X}'$ in ascending order while the $\mathbf{Idx}$ is the index vector of $\mathbf{V}$. The plain image pixels are then permutated according to the index value.

7.2.2 Diffusion stage

To impose the avalanche effect in the encryption scheme, the pixel value is modified through the diffusion process. Inspired by Hua et al. (2018), a random matrix $\mathbf{R}$ is inserted to the permutated image using the modular addition to eliminate the linear relationship between plain and cipher images. Firstly, a sequence $\mathbf{Y}' = \{y'(i)\}_{i=0}^{MN}$ is generated by using the sequence $\mathbf{Y}$ from Equation (7.1) through the following quantization.

$$y'(i) = \text{floor}([\text{abs}(y(i)) - \text{floor}(y(i))] \times 10^{15}) \bmod 256. \tag{7.7}$$

The 1D array of $\mathbf{Y}'$ is then reshape into a $M \times N$ matrix $\mathbf{R}$ and the modular addition is applied on the permutated matrix $\mathbf{P}'$ from Equation (7.6) and $\mathbf{R}$.

$$\mathbf{R} = \text{reshape}(\mathbf{Y}', M, N), \tag{7.8}$$

$$\mathbf{P}'' = (\mathbf{P}' + \mathbf{R}) \bmod 256, \tag{7.9}$$

where $\mathbf{P}'' = \{p''(i)\}_{i=0}^{MN}$.

Besides, the pixel values are further diffused by using a nonlinear equation. The nonlinear equation is formed by the combination of exclusively-OR and modular addition of two random masks, permutated pixels and previous cipher pixels. The random masks are the quantized sequences $\mathbf{Z}'$ and $\mathbf{W}'$ given by

$$z'(i) = \text{floor}([\text{abs}(z(i)) - \text{floor}(z(i))] \times 10^{15}) \bmod 256. \tag{7.10}$$

$$w'(i) = \text{floor}([\text{abs}(w(i)) - \text{floor}(w(i))] \times 10^{15}) \bmod 256. \tag{7.11}$$

Then, the nonlinear equation is represented by

$$\begin{aligned} c(i) = \; & [(z'(i)+w'(i)) \bmod 256] \oplus (z'(i)+[(p''(i)+w'(i)) \\ & \bmod 256 \oplus c(i-1)]) \bmod 256, \end{aligned} \tag{7.12}$$

where $c(i)$ is the i^{th} pixels of the cipher image $\mathbf{C} = c(i)_{i=0}^{MN}$.

7.2.3 Encryption algorithm

The detailed encryption process is given as follows:

Input The plain image $\mathbf{P}$ with size of $M \times N$, Secret keys (a,b,c,d,e,h,k), (x_0, y_0, z_0, w_0), and N_0.

Output Cipher image $\mathbf{C}$.

1. Generate $\mathbf{K}$ and updated initial conditions according to Equation (7.2) and (7.3).

2. Iterate hyperchaotic system in Equation (7.1) for $N_0 + MN$ times and discard the first N_0 elements to avoid harmful transient effect. Four chaotic sequences are obtained $\mathbf{X} = \{x(i)\}_{i=0}^{MN}$, $\mathbf{Y} = \{y(i)\}_{i=0}^{MN}$, $\mathbf{Z} = \{z(i)\}_{i=0}^{MN}$ and $\mathbf{W} = \{w(i)\}_{i=0}^{MN}$.

3. Obtain sequence $\mathbf{X}'$ using Equation (7.4) and shuffle the plain image pixels according to the permutation vector given in Equations (7.5) and (7.6).

4. Obtain sequence $\mathbf{Y}'$ using Equation (7.7) and reshape the random matrix $\mathbf{R}$ by using Equation (7.8). Insert the random matrix $\mathbf{R}$ into the permutated image using Equation (7.9).

5. Obtain two random masks $\mathbf{Z}'$ and $\mathbf{W}'$ using Eqs. (7.10) and (7.11) and perform diffusion using Equation (7.12) to get cipher image $\mathbf{C}$.

7.2.4 Decryption algorithm

The detailed decryption process is given as follows:

Input The cipher image **C**, Secret keys (a,b,c,d,e,h,k), (x_0,y_0,z_0,w_0), and N_0.

Output The recovered plain image **P**.

1. Use updated initial conditions to generate hyperchaotic system in Equation (7.1) for N_0+MN times and discard the first N_0 elements to get rid of the harmful transient effect and obtain four chaotic sequences $\mathbf{X}=\{x(i)\}_{i=0}^{MN}$, $\mathbf{Y}=\{y(i)\}_{i=0}^{MN}$, $\mathbf{Z}=\{z(i)\}_{i=0}^{MN}$ and $\mathbf{W}=\{w(i)\}_{i=0}^{MN}$.
2. Obtain quantized chaotic sequences $\mathbf{Z}'$ and $\mathbf{W}'$ according to Eqs. (7.10) and (7.11).
3. Obtain $\mathbf{P}''$ using the following the operation.
   $p''(i)=((((c(i)\oplus((z'(i)+w'(i)) \bmod 256))-z'(i)) \bmod 256)\oplus c(i-1))-w'(i) \bmod 256.$
4. Get sequence $\mathbf{Y}'$ and matrix $\mathbf{R}$ by using Eqs. (7.7) and (7.8) and obtain $\mathbf{P}'=(\mathbf{P}''-\mathbf{R}) \bmod 256$.
5. Recover plain image by performing $\mathbf{P}(\mathbf{Idx})=\mathbf{P}'$.

7.3 Security analysis

The simulations for image encryption algorithm were implemented in MATLAB R2019a, using a personal computer with Intel® Core™ i5-8250 CPU @ 1.60GHz, 8 GB memory and a Windows 10 operating system. To evaluate the performance of our proposed encryption scheme, we selected a 256×256 plain image titled 'Airplane' from the USC-SIPI image database. The secret keys used in the test are $a = 5$, $b = 20$, $c = 1$, $d = 0.1$, $e = 20.6$, $h = 1$, $k = 0.1$, $x_0 = 3.2$, $y_0 = 8.5$, $z_0 = 3.5$ and $w_0 = 2.0$. To test the security of the proposed method, the key and subkey space, ability to resist statistical and differential attacks, information entropy, and secret key sensitivity are tested in the experiments.

7.3.1 Key and subkey space

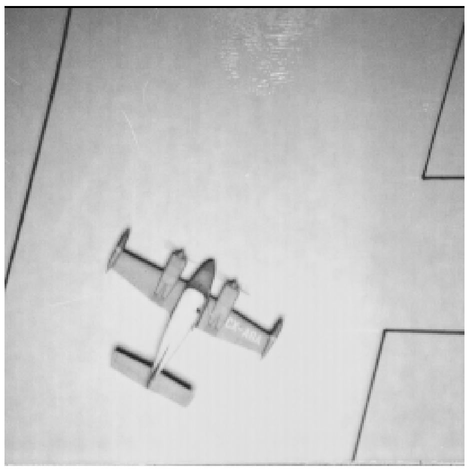
For a secure image encryption, the key space must be large enough to withstand the brute force attack. The secret keys are the initial conditions (x_0, y_0, z_0, w_0) and control parameters of hyperchaotic system (a, b, c, d, e, h, k), and the digest of SHA-256 $\mathbf{K} = k_1, k_2, \ldots, k_{32}$. According to IEEE standard for floating point (Rajaraman, 2016), the computation precision of floating-point number is around 10^{-15}. So, the key space is $10^{15\times 11} \times 2^{256} \approx 2^{342.37}$.

According to Yap et al. (2016), the encryption scheme is breakable if the subkeys can be recovered by the attacker easily. In the proposed scheme, the plain image can be recovered if the (a, b, c, d, e, h, k) and (x_0', y_0', z_0', w_0') are known. There exist $10^{15\times 7}$ possible values for (a, b, c, d, e, h, k). On the other hand, the possible values of (x_0', y_0', z_0', w_0') depend on (x_0, y_0, z_0, w_0) and the exclusively-or of k_i, for $1 \leq i \leq 32$. To obtain x_0' in Equation (7.3), we require x_0 and $k_1 \oplus k_2 \oplus \cdots \oplus k_8$, and their possible values are 10^{15} and 2^8, respectively. Same argument applies on y_0', z_0', and w_0'. Thus, the effective key space of this scheme is $10^{15\times 7} \times (10^{15} \times 2^8)^4 \approx 2^{254.57}$. Since it is still greater than 2^{100}, therefore it is sufficient to resist brute-force attack (Alvarez and Li, 2006).

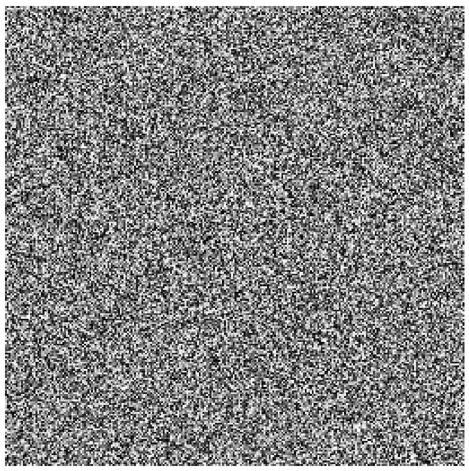
7.3.2 Statistical attack

Histogram and chi-square test

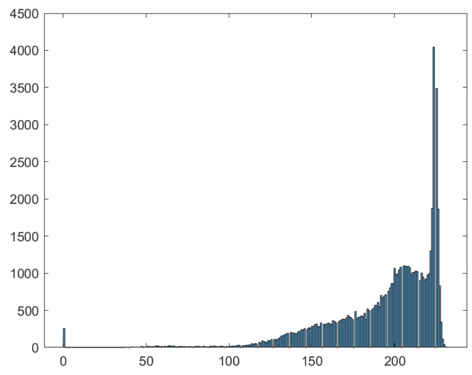
A secure image encryption must ensure that the cipher image is uniformly distributed. Histogram is plotted to show the distribution of the pixel intensity of the image. Figures 7.2a and 7.2b show the plain and cipher images whereas their respective histogram are plotted in Figures 7.2c and 7.2d. It is obvious that the pixel values of the plain image are distributed in an unbalanced mode while the pixel values of the cipher image are uniformly distributed.



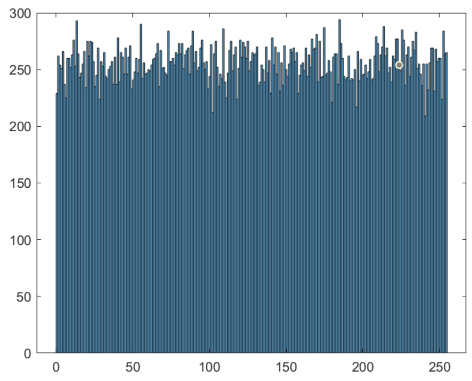
(a)



(b)



(c)



(d)

Figure 7.2: Histogram Analysis. (a) Plain image. (b) Cipher Image. (c) Histogram of Plain image. (d) Histogram of Cipher image.

To further verify the uniformity of the pixel distribution of the cipher image, chi-square (χ^2) test is carried out using

$$\chi^2 = \sum_{i=1}^{L} \frac{(o(i) - m(i))^2}{m(i)},$$

where $o(i)$ and $m(i)$ denote the frequency of observations and expected frequency at the i^{th} interval, respectively, and L is the maximum level of grayscale image, i.e., 256 in this scheme. The smaller the χ^2 value, the closer the distribution of cipher images to the uniform distribution. The result shows that the χ^2 value is 274.875 which is lower than the critical value of 293 under significance level of 5%. Therefore, the histogram of the cipher image is uniform.

Correlation analysis

Image data have a high correlation to the adjacent pixels in different directions i.e., horizontal, vertical and diagonal directions. Attacker could exploit this feature to retrieve the information of the images. To test correlation between the adjacent pixels of the encrypted image, coefficient of adjacent pixels ρ_{XY} is calculated by

$$\begin{aligned}
\rho_{XY} &= \frac{Cov_{XY}}{\sigma_X \sigma_Y}, \\
Cov_{XY} &= \frac{1}{N}\sum_{i=1}^{N}(x_i - E(X))(y_i - E(Y)), \\
E(X) &= \frac{1}{N}\sum_{i=1}^{N} x_i, & E(Y) &= \frac{1}{N}\sum_{i=1}^{N} y_i \\
\sigma_X^2 &= \frac{1}{N}\sum_{i=1}^{N}(x_i - E(X))^2, & \sigma_Y^2 &= \frac{1}{N}\sum_{i=1}^{N},(y_i - E(Y))^2
\end{aligned}$$

where X and Y are two adjacent pixels and N is the total number of duplets (X,Y) from the image. From Table 7.1, it shows that the correlation values for the cipher image are close to zero, which means that the pixels of the cipher

Table 7.1: Correlation Analysis of Plain and Cipher Images

Image	Horizontal	Vertical	Diagonal
Plain image	0.9571	0.9366	0.8927
Cipher image	-0.0093	0.0026	0.0055

image are not correlated to each other.

7.3.3 Differential attack

The commonly used statistical tests to measure the strength of the underlying encryption scheme against differential attack are number of pixels change rate (NPCR) and unified average change intensity (UACI). They are represented by

$$NPCR(C_1,C_2) = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} F(i,j)}{N \times M} \times 100\%,$$

$$F(i,j) = \begin{cases} 0, & \text{if } C_1(i,j) = C_2(i,j), \\ 1, & \text{if } C_1(i,j) \neq C_2(i,j), \end{cases}$$

$$UACI(C_1,C_2) = \frac{1}{MN}\sum_{i=1}^{M}\sum_{i=1}^{N} \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} |C_1(i,j) - C_2(i,j)|}{L-1} \times 100\%,$$

where L is the largest allowable pixel value in the image, while C_1 and C_2 are two cipher images with one pixel difference. In the experiment, the upper-left pixel of the image is added by one pixel to test on the differential attack. As mentioned in (Wu et al., 2011), the ideal expected values of NPCR and UACI in a grayscale image should be 99.6094% and 33.4635%, respectively. From the experiment, the NPCR and UACI values are 99.6307% and 33.4636%, respectively, which are higher than the ideal values. This shows that our proposed scheme has good avalanche effect and can resist to differential attack.

7.3.4 Information entropy

Information entropy is a test to measure the randomness and the distribution of the image pixels. It can be calculated using

$$H(m) = -\sum_{i=0}^{L-1} P[m(i)] \log_2 P[m(i)],$$

where $P[m(i)]$ is the probability of occurrence of $m(i)$, L is the maximum level of grayscale image, i.e., 256 in this scheme. The maximum value of entropy is 8. The closer the value near to 8, the higher the randomness of the pixels in the image. In this experiment, the entropy of cipher image is 7.9975, which is close to 8, so the randomness of the cipher image is satisfactory.

7.3.5 Key sensitivity

This test measures the sensitivity of cipher image to a tiny change in secret key. A robust algorithm should be able to secure from the attack with a slight change in the secret key. The attacker cannot break the algorithm and obtain the useful information by using a similar key. The key sensitivity tests can be conducted in two methods: (1) a different encrypted image is produced with the altered key, and (2) the encrypted image cannot be decrypted with a slightly altered key.

For the first method, to determine which secret key x_0, y_0, z_0, or w_0 has the highest sensitivity, we conducted a test by adding 10^{-15} to each key individually while keeping the other keys unchanged. Our comparisons revealed that y_0 has the greatest effect on the cipher. Specifically, the image encrypted using $y_0 + 10^{-15}$ exhibited the highest difference ratio compared to the image encrypted

Table 7.2: Difference ratio between the encrypted image using the original key and the updated key.

Updated Key	$x_0 + 10^{-15}$	$y_0 + 10^{-15}$	$z_0 + 10^{-15}$	$w_0 + 10^{-15}$
Difference Ratio	0.995681	0.996490	0.996124	0.996094

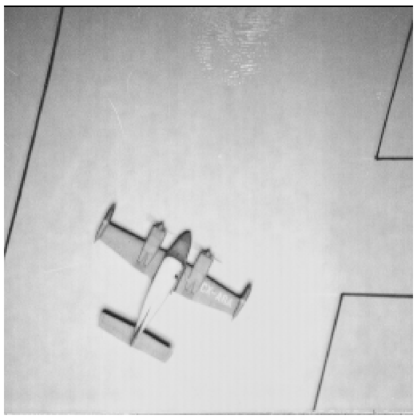
with the original key. The difference ratios for each updated key are summarized in Table 7.2. For graphical illustration, we utilize two keys: key_1 $(x_0 = 3.2, y_0 = 8.5, z_0 = 3.5$ and $w_0 = 2.0)$ and key_2 $(x_0 = 3.2, y_0'' = 8.5 + 10^{-15}, z_0 = 3.5$ and $w_0 = 2.0)$. We encrypt the same plain image using both key_1 and key_2, resulting in different encrypted images (refer to Figures 7.3b and 7.3c). The difference ratio between these two encrypted images, as shown in Figure 7.3d, is 0.99649. This indicates that 99.649% of the pixels between the two images are different. Therefore, the encrypted image produced by a slightly altered key is entirely different and cannot be exploited by an attacker.

In the second method, we use key_1 to encrypt the plain image (see Figures 7.3a and 7.3b). Then, both key_1 and key_2 are used to decrypt the cipher image. Both key_1 and key_2 are then employed to decrypt the encrypted image. As shown in Figures 7.3e and 7.3f, the original image can only be recovered using key_1. The image decrypted with key_2 is completely unrecognizable.

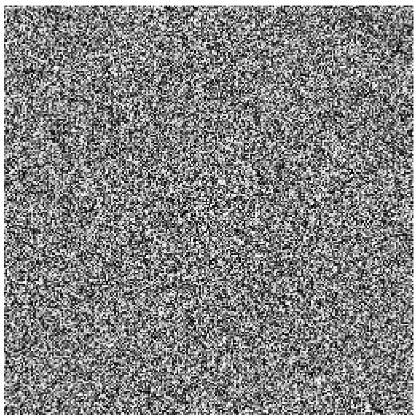
7.3.6 Comparison of the test performance

The performance of the proposed scheme is compared with the encryption schemes presented by Boriga et al. (2014); Zhang et al. (2016); Biswas et al. (2015); Khan (2015) and the values are summarized in the Table 7.3.

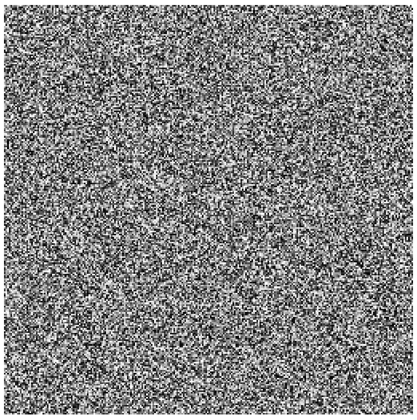
Our proposed scheme has the largest key space which indicates that it is the safest against the brute force attack. Even though the correlation of the proposed scheme is not the best, it is very close to 0. That means the cipher



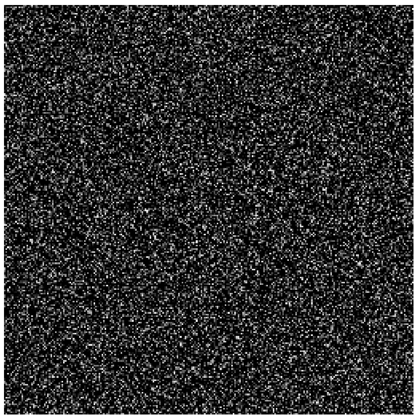
(a)



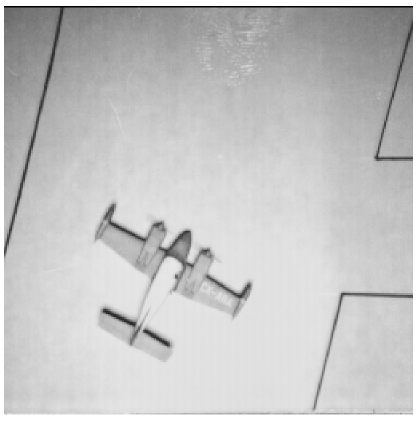
(b)



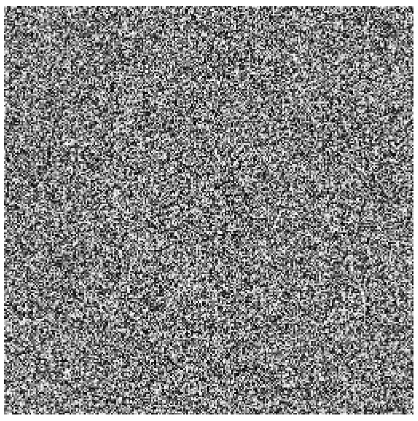
(c)



(d)



(e)



(f)

Figure 7.3: Key Sensitivity Analysis. (a) Plain image. (b) Cipher Image using key_1. (c) Cipher Image using key_2. (d) Difference between Figures 7.3b and 7.3c (e) Recovered image using key_1. (f) Recovered image using key_2.

Table 7.3: Comparison of Test Performance

Test	Proposed	Boriga et al.	Zhang et al.	Biswas et al.	Khan
Key space	$2^{254.57}$	2^{248}	2^{159}	2^{448}	2^{159}
Correlation:					
Horizontal	-0.0093	0.001587	-0.004223	0.0027	0.0107
Vertical	-0.0026	0.014706	0.00055	0.0019	0.0141
Diagonal	0.0055	0.002381	-0.003665	0.0070	0.0097
NPCR	99.6307%	99.27%	99.6155%	99.676%	99.6124%
UACI	33.4636%	33.22%	33.4988%	33.422%	33.4591%
Information Entropy	7.9975	7.999282	7.9992495	7.9988	7.9972

pixels are not correlated to the adjacent pixels. The NPCR and UACI scores in our proposed scheme is higher than than the ideal values, i.e. 99.6307% and 33.4636%, respectively. Thus, it is good in resisting against the differential attack. Lastly, the information entropy of our proposed scheme is not as good as the other. However, it is very close to the 8, therefore the randomness of the proposed scheme is still satisfactory.

7.4 Summary

This chapter presents a new image encryption scheme based on the hyperchaotic system and SHA-2 algorithm. The hyperchaotic system has a better chaotic behavior over the lower-dimensional chaotic system in terms of ergodicity, sensitivity to the initial condition and control parameters, randomness and structural complexity. Furthermore, the use of SHA-256 hash function in modifying the initial conditions of the hyperchaotic system highly enhances the sensitivity of the cipher to the change of plain image. The nonlinear equation used in the diffusion process also introduces a good avalanche effect in the encryption scheme. To test the security level of the proposed scheme, a series of experiments have been conducted. All the numerical results demonstrate that our proposed scheme has good security performance and thus it is suitable for image encryption.

CHAPTER 8

CROSS-PLANE COLOR IMAGE ENCRYPTION BASED ON TWO-DIMENSIONAL SINE-HENON MAP AND GENETIC ALGORITHM

Building on the cryptanalysis conducted in Chapters 3 and 4, this chapter presents a new image encryption scheme that addresses the weaknesses identified in previous methods suitable for color images. Chapter 3 examined Biswas et al.'s encryption scheme, which demonstrated low sensitivity to plaintext changes, violating fundamental cryptographic principles such as strong diffusion and nonlinearity. Similarly, Chapter 4 investigated Ping et al.'s encryption method, which relies heavily on the Henon map for both permutation and diffusion. However, due to the dynamical degradation of the Henon map under certain conditions, the encryption exhibited weak diffusion effects, making it vulnerable to chosen-plaintext attacks. Chapters 5 and 6 introduced the cascading method to improve the chaotic properties of maps. Motivated by these approaches, we propose a novel chaotic map that integrates the sine trigonometric function with the Henon map to enhance its chaotic behavior.

Unlike the grayscale image encryption proposed in Chapter 7, this chapter focuses on dynamically encrypting the cross-planes of color images to maximize security. To further enhance the sensitivity of the cipher to plaintext changes, we incorporate the SHA-256 hash function, ensuring that even minor modifications in the input image produce significant alterations in the encryption process. Additionally, we integrate genetic algorithms that leverage the intrinsic characteristics of image bit distributions, applying mutation and crossover operations in a dynamic order. A novel uniform crossover method is introduced to improve randomness. Lastly, pixel-level diffusion is implemented to enhance the avalanche effect, strengthening the overall security of the encryption scheme against cryptanalysis.

8.1 Introduction

The majority of proposed image encryption schemes work at the pixel level. Most of these schemes encounter an issue where the pixel value and histogram statistics remain unchanged after the permutation process, primarily due to insensitivity to tiny change of the plain-image. To overcome this problem, many researchers incorporate bit-level operations into their designs, which involve studying the bit distributions within each pixel. Furthermore, a significant number of them combine this technique with DNA coding (Rehman et al., 2019; Akkasaligar and Biradar, 2020). When performing the permutation and diffusion process, intrinsic features of the image are taken into consideration. This process involves dividing the image into two blocks, namely those containing the most significant bits (MSB) and least significant bits (LSB) respectively. Different treatments are given to MSB and LSB blocks, whereby the MSB block that carries around 94.12% of information should be given more attention. In the diffusion process, bit-level encryption is susceptible to chosen-plaintext attacks. Therefore, we continue to employ pixel-level diffusion after bit-level encryption.

Genetic algorithm was firstly introduced by Holland (1975, 1992) in encryption algorithm. This technique imitates the natural evolution and selection process. To apply genetic algorithm in image encryption, the genetic information is substituted by the pixel or bit values of an image. There are two genetic operators in this technique: crossover and mutation. For a new sequence to be generated, a pair of parent sequences and the crossover region are chosen. The information in between the crossover region of parent sequences are swapped to produce the child sequences. Mutation operator is a process to change the information of the selected sequences through the flipping of the bit values. Biswas et al. (2015) presented an image encryption algorithm based on genetic operations for wireless sensor network but it was found insecure against the known plaintext attack (Wong, Yap, Wong, Phan and Goi, 2020). Mozaffari (2018) proposed a grayscale image encryption algorithm using the crossover and mutation operations to perform the bitplane permutation and substitution processes. Zhang, He, Li and Wang (2020) proposed an color image encryption by converting each color plane into a one-dimensional sequence and applying the genetic algorithms on each color plane separately. The common weaknesses of these image encryption algorithms are the secret keys for the mutation and crossover are independent of the plain image, resulting the algorithms are vulnerable to the plaintext-like attacks.

In this chapter, we present a color image encryption based on the two-dimensional sine-henon chaotic map and genetic algorithm. We also adopt the cross-plane selective encryption method to improve the encryption power and security level.

Organization: The remainder of this chapter is organised as follows. In the next section, we recall the operations of the genetic algorithm. We then presents a new two-dimensional chaotic map based on sine and henon maps to generate the pseudorandom sequences for encryption in Subsection 8.3. In Section 8.4, we present an image encryption scheme that composes both bit-level encryption and pixel-level diffusion. In Section 8.5, we present the security analysis on the key space, key sensitivity and resistance to chosen plaintext attack. Section 8.6 concludes the chapter.

8.2 Genetic Algorithm

Genetic algorithm is a technique that emulates the principles of natural selection and genetics observed in biological evolution. It involves two fundamental operations: crossover and mutation. Crossover is a process of exchanging the selected part of two parent bit strings to produce the child bit strings. The crossover technique includes single-point, two-point and multi-point crossover Wong, Yap, Wong, Phan and Goi (2020). On the other hand, mutation introduces random changes to the offspring population.

8.2.1 One-point crossover

Single-point crossover is a process where single point is chosen and the parts before or after of the chosen point are exchanged between two parent bit strings. Let $A = (a_1, a_2, \ldots, a_n)$ and $B = (b_1, b_2, \ldots, b_n)$ be two parent bit strings with size of n, and crossover points $f \in \{1, \ldots, n\}$ and $g \in \{1, \ldots, n\}$ are the index of the parent bit strings to be swapped. Let $A' = (a'_1, a'_2, \ldots, a'_n)$ and $B' = (b'_1, b'_2, \ldots, b'_n)$ be the corresponding child bit strings. When the crossover points f and g are equal, we apply the one-point crossover on the parent bit strings A and B. If $f > \frac{n}{2}$, then the first bit until the f^{th} bit of the parent bit strings will be interchanged, the crossover process $(A', B') = Crossover_1(A, B, f, g)$ is given in Equation (8.1). Refer to Figure 8.1a for graphical illustration.

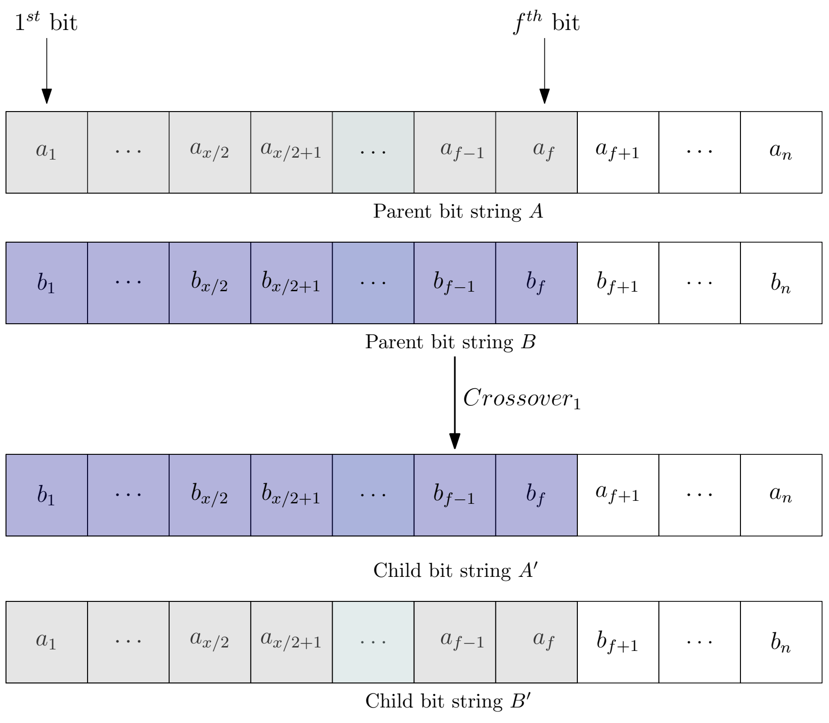
$$a'_i = \begin{cases} b_i, & \text{for } 1 \le i \le f, \\ a_i, & \text{for } f < i \le n. \end{cases}$$
$$b'_i = \begin{cases} a_i, & \text{for } 1 \le i \le f, \\ b_i, & \text{for } f < i \le n. \end{cases} \tag{8.1}$$

If $f \le \frac{n}{2}$, then the $(f+1)^{th}$ bit until the last bit of the parent bit strings will be interchanged, refer to Figure 8.1b for the graphical demonstration. The child bit strings $(A', B') = \mathit{Crossover_1}(A, B, f, g)$ can be obtained by

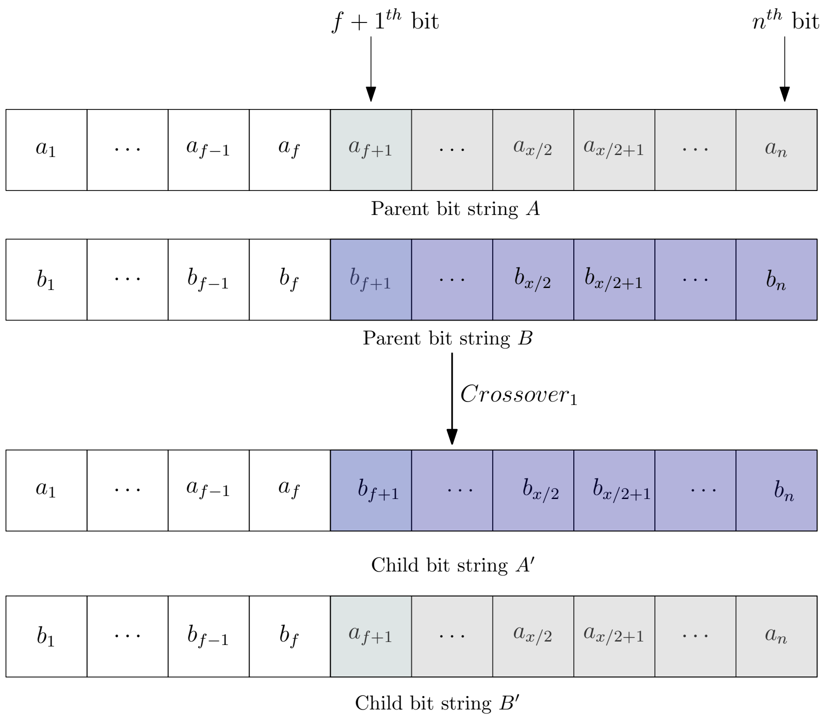
$$a'_i = \begin{cases} b_i, & \text{for } f < i \le n, \\ a_i, & \text{for } 1 \le i \le f. \end{cases}$$
$$b'_i = \begin{cases} a_i, & \text{for } f < i \le n, \\ b_i, & \text{for } 1 \le i \le f. \end{cases} \tag{8.2}$$

8.2.2 Two-point crossover

Similar to one-point crossover, the selected part in between two points of two parent bit strings are exchanged. Let f and g are two crossover points. When f and g are different, two-point crossover will be applied the parent bit strings A and B. If $f < g$, the bit values in between f^{th} until g^{th} positions of the two parent bit strings are interchanged and produce the child bit strings $A' = (a'_1, a'_2, \ldots, a'_n)$ and $B' = (b'_1, b'_2, \ldots, b'_n)$. The equation $(A', B') = \mathit{Crossover_2}(A, B, f, g)$ are shown in Equation (8.3), refer to Figure 8.2a for graphical illustration. Equation (8.3) also applies for $f > g$, where the graphical representation of the process is


1^{st} bit
f^{th} bit
a_1 $\cdots$ $a_{x/2}$ $a_{x/2+1}$ $\cdots$ a_{f-1} a_f a_{f+1} $\cdots$ a_n
Parent bit string A
b_1 $\cdots$ $b_{x/2}$ $b_{x/2+1}$ $\cdots$ b_{f-1} b_f b_{f+1} $\cdots$ b_n
Parent bit string B
$Crossover_1$
b_1 $\cdots$ $b_{x/2}$ $b_{x/2+1}$ $\cdots$ b_{f-1} b_f a_{f+1} $\cdots$ a_n
Child bit string A'
a_1 $\cdots$ $a_{x/2}$ $a_{x/2+1}$ $\cdots$ a_{f-1} a_f b_{f+1} $\cdots$ b_n
Child bit string B'

(a)


$f+1^{th}$ bit
n^{th} bit
a_1 $\cdots$ a_{f-1} a_f a_{f+1} $\cdots$ $a_{x/2}$ $a_{x/2+1}$ $\cdots$ a_n
Parent bit string A
b_1 $\cdots$ b_{f-1} b_f b_{f+1} $\cdots$ $b_{x/2}$ $b_{x/2+1}$ $\cdots$ b_n
Parent bit string B
$Crossover_1$
a_1 $\cdots$ a_{f-1} a_f b_{f+1} $\cdots$ $b_{x/2}$ $b_{x/2+1}$ $\cdots$ b_n
Child bit string A'
b_1 $\cdots$ b_{f-1} b_f a_{f+1} $\cdots$ $a_{x/2}$ $a_{x/2+1}$ $\cdots$ a_n
Child bit string B'

(b)

Figure 8.1: One-point crossover process

shown in Figure 8.2b.

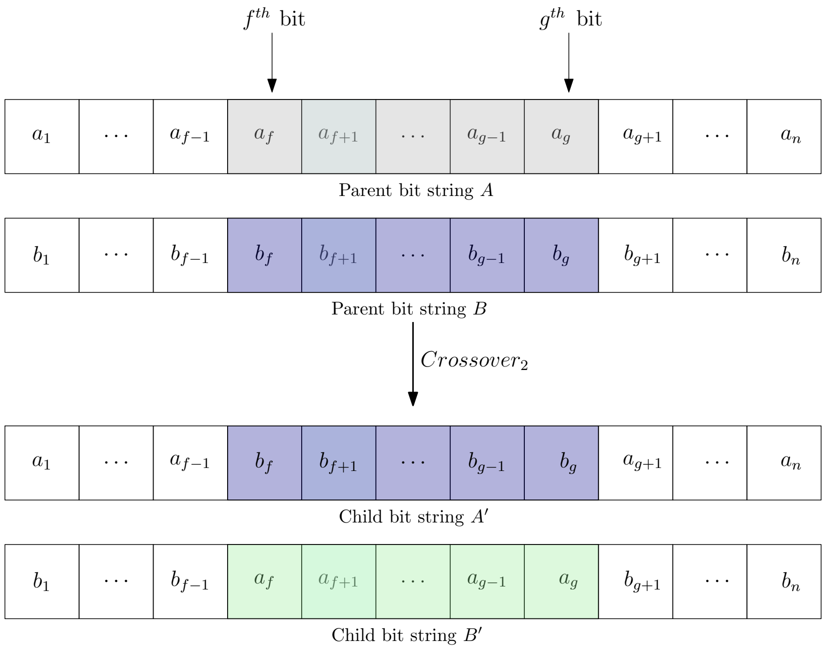
$$
\begin{aligned}
a_i' &= \begin{cases} b_i, & \text{for } f \le i \le g, \\ a_i, & \text{for } i < f \text{ or } i > g. \end{cases} \\
b_i' &= \begin{cases} a_i, & \text{for } f \le i \le g, \\ b_i, & \text{for } i < f \text{ or } i > g. \end{cases}
\end{aligned}
\tag{8.3}
$$

8.2.3 Uniform crossover

Uniform crossover is a process where the child bits are produced from either parent with equal probability. Let $A = (a_1, a_2, \dots, a_n)$ and $B = (b_1, b_2, \dots, b_n)$ be the parent bit strings and $C = (c_1, c_2, \dots, c_n)$ and $D = (d_1, d_2, \dots, d_n)$ be two binary strings with length n. In this chapter, the bits of A and B will be interchanged if the bits of the corresponding positions in C and D are equal, i.e., $c_i = d_i$ for $i \in \{1, 2, \dots, n\}$. The child bit strings $A' = (a_1', a_2', \dots, a_n')$ and $B' = (b_1', b_2', \dots, b_n')$ of the uniform crossover are produced based on $(A', B') = Crossover_u(A, B, C, D)$, where

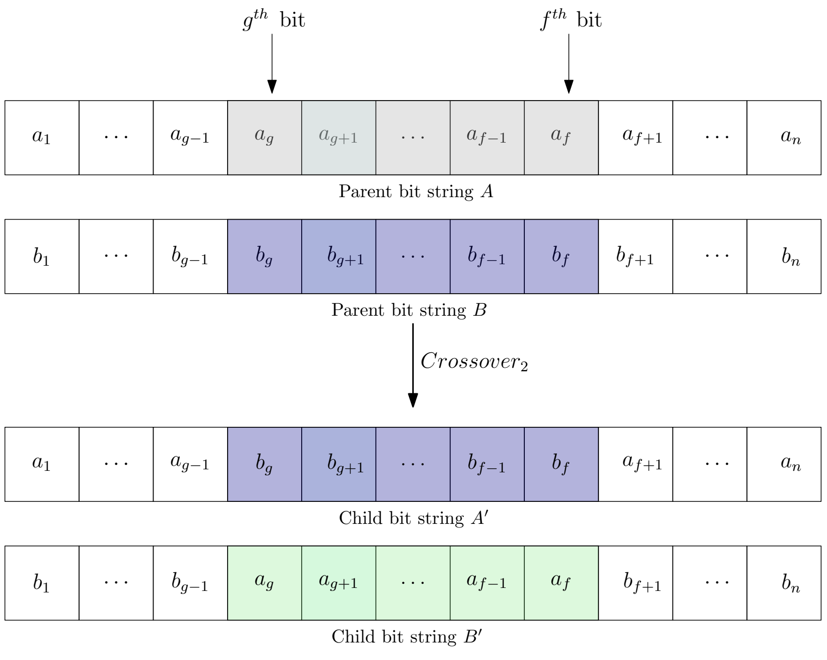
$$
\begin{aligned}
a_i' &= \begin{cases} b_i, & \text{for } c_i = d_i, \\ a_i, & \text{for } c_i \ne d_i. \end{cases} \\
b_i' &= \begin{cases} a_i, & \text{for } c_i = d_i, \\ b_i, & \text{for } c_i \ne d_i. \end{cases}
\end{aligned}
\tag{8.4}
$$

Refer to the example given in Figure 8.3, as the values of bits at position $2, 3, 4$ and 7 of binary string C and D are the same, the bits at the corresponding positions of parent bit strings A and B are interchanged.



f^{th} bit
g^{th} bit
a_1 $\cdots$ a_{f-1} a_f a_{f+1} $\cdots$ a_{g-1} a_g a_{g+1} $\cdots$ a_n
Parent bit string A
b_1 $\cdots$ b_{f-1} b_f b_{f+1} $\cdots$ b_{g-1} b_g b_{g+1} $\cdots$ b_n
Parent bit string B
$Crossover_2$
a_1 $\cdots$ a_{f-1} b_f b_{f+1} $\cdots$ b_{g-1} b_g a_{g+1} $\cdots$ a_n
Child bit string A'
b_1 $\cdots$ b_{f-1} a_f a_{f+1} $\cdots$ a_{g-1} a_g b_{g+1} $\cdots$ b_n
Child bit string B'

(a)



g^{th} bit
f^{th} bit
a_1 $\cdots$ a_{g-1} a_g a_{g+1} $\cdots$ a_{f-1} a_f a_{f+1} $\cdots$ a_n
Parent bit string A
b_1 $\cdots$ b_{g-1} b_g b_{g+1} $\cdots$ b_{f-1} b_f b_{f+1} $\cdots$ b_n
Parent bit string B
$Crossover_2$
a_1 $\cdots$ a_{g-1} b_g b_{g+1} $\cdots$ b_{f-1} b_f a_{f+1} $\cdots$ a_n
Child bit string A'
b_1 $\cdots$ b_{g-1} a_g a_{g+1} $\cdots$ a_{f-1} a_f b_{f+1} $\cdots$ b_n
Child bit string B'

(b)

Figure 8.2: Two-point crossover process

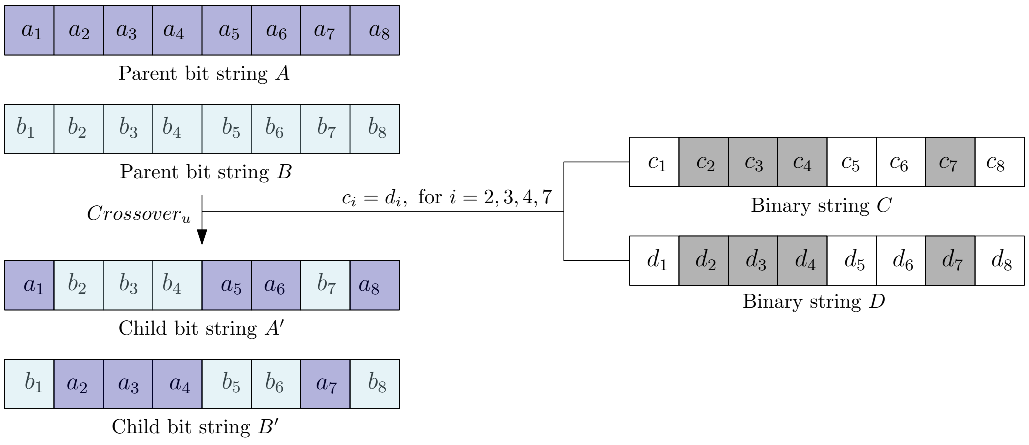

a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8
Parent bit string A
b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8
Parent bit string B
$Crossover_u$
$c_i = d_i$, for $i = 2, 3, 4, 7$
c_1 c_2 c_3 c_4 c_5 c_6 c_7 c_8
Binary string C
d_1 d_2 d_3 d_4 d_5 d_6 d_7 d_8
Binary string D
a_1 b_2 b_3 b_4 a_5 a_6 b_7 a_8
Child bit string A'
b_1 a_2 a_3 a_4 b_5 b_6 a_7 b_8
Child bit string B'

Figure 8.3: Uniform crossover process

8.2.4 Mutation

Mutation is commonly known as a negation operator that changes one or multiple bits in a given bit string (Hassan and Abuhaiba, 2011). Let $G = (g_1, g_2, \ldots, g_n)$ be a n-bit string. Let $k \in \{0, 1, \ldots, n\}$ be the mutation. Let $Mutation(G, k)$ be the function of mutation, where G and k are two parameters for this function. The output of the function, string $H = (h_1, h_2, \ldots, h_n)$ is obtained by flipping every k^{th} to n^{th} bits by one bit. If $k = 0$, $G = H$. If $k \neq 0$, then

$$h_j = \begin{cases} g_j, & \text{for } 1 \leq j \leq k-1, \\ 1 - g_j, & \text{for } k \leq j \leq n. \end{cases} \tag{8.5}$$

Figure 8.4 shows the graphical illustration of the mutation process.

8.3 Proposed chaotic map

The chaotic map plays a crucial role in image encryption by generating pseudorandom sequences, and its dynamical performance significantly

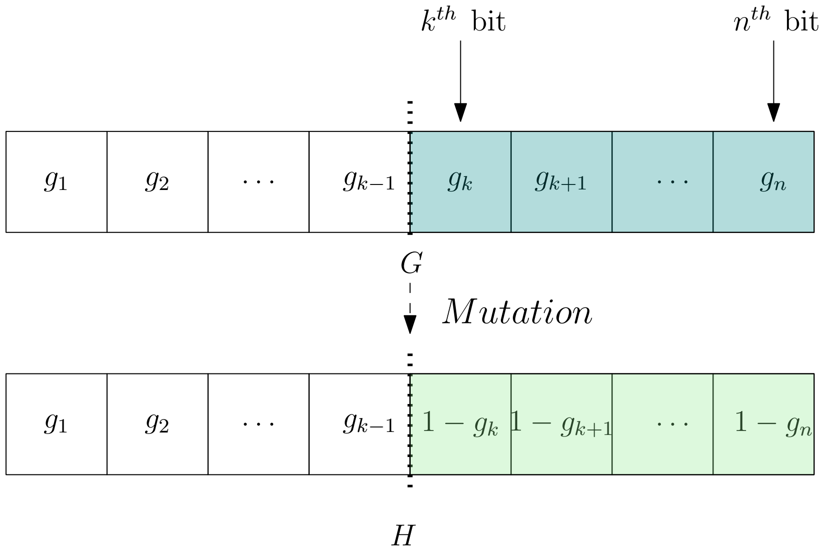


Figure 8.4: Mutation process

influences the security of encryption. Many researchers favor one-dimensional chaotic maps due to their simple structures, which facilitate high efficiency. However, they suffer from drawbacks such as limited and discontinuous chaotic ranges. The control parameters of these maps serve as the secret keys, but their small chaotic ranges make the key space becomes small. High-dimensional chaotic maps are applied in cryptographic applications because of their broad chaotic ranges and complex structures, despite requiring higher computational cost. The introduction of two-dimensional chaotic maps addresses the limitations encountered in both one-dimensional and high-dimensional chaotic maps. By offering a balance between efficiency and performance, two-dimensional chaotic maps are widely adopted in image encryption.

8.3.1 2D-SHCM

In order to show the chaotic behavior of a dynamical system, it is necessary to demonstrate that the chaotic map is bounded. The sine function is a trigonometric function generating the outputs that are bouded within the interval [-1,1]. Unlike the regular Henon map, the 2D-SHCM can leverage the entire range for parameters *a* and *b* because the sine function itself exhibits chaotic behavior when *k* is large, and even slight difference in the output of the seed maps can lead to significant difference in the overall chaotic system. Moreover, the cascade system further modifies the chaotic states of the Henon map after each iteration. The dynamical system of 2D-SHCM is represented mathematically by

$$\begin{aligned} x_{i+1} &= \sin(\pi(1+y_i-a(x_i+k)^2)), \\ y_{i+1} &= \sin(\pi(b(x_i+k)), \end{aligned} \tag{8.6}$$

where $\{x_i, y_i\}_{i=0}^{\infty} \in [-1,1]$ are the state variables, *a* and *b* are control parameters and *k* is a constant.

The proposed 2D-SHCM improve the chaotic performance of the sine and henon maps. To demonstrate its strength, we perform evaluatuation on its chaotic performance and compare it with some recently proposed 2D chaotic maps. The evaluations are performed using phase diagram, bifurcation diagram, and state mapping network. A phase diagram visualizes the dynamical behavior of a chaotic map within the state space over time. Bifurcation diagram illustrates the transition of chaotic map from periodic to chaotic behavior as the parameters change. Additionally, the state mapping network serves as a functional graph to study the period and cycle distribution of the chaotic map in the digital domain.

8.3.2 Phase diagram

Phase diagram serves as a tool for observing the chaotic trajectories of the proposed maps in phase space. As illustrated in Fig. 8.5, the attractor of

2D-SHCM exhibits greater complexity compared to those of 2D-CLSS (Teng et al., 2022) and 2D-SCS (Hua et al., 2019). Analysis of these diagrams reveals that the distribution region of 2D-SHCM is significantly larger than that of 2D-CLSS and 2D-SCS. It means that 2D-SHCM has better ergodicity and randomness, making it more resilient against various forms of attacks.

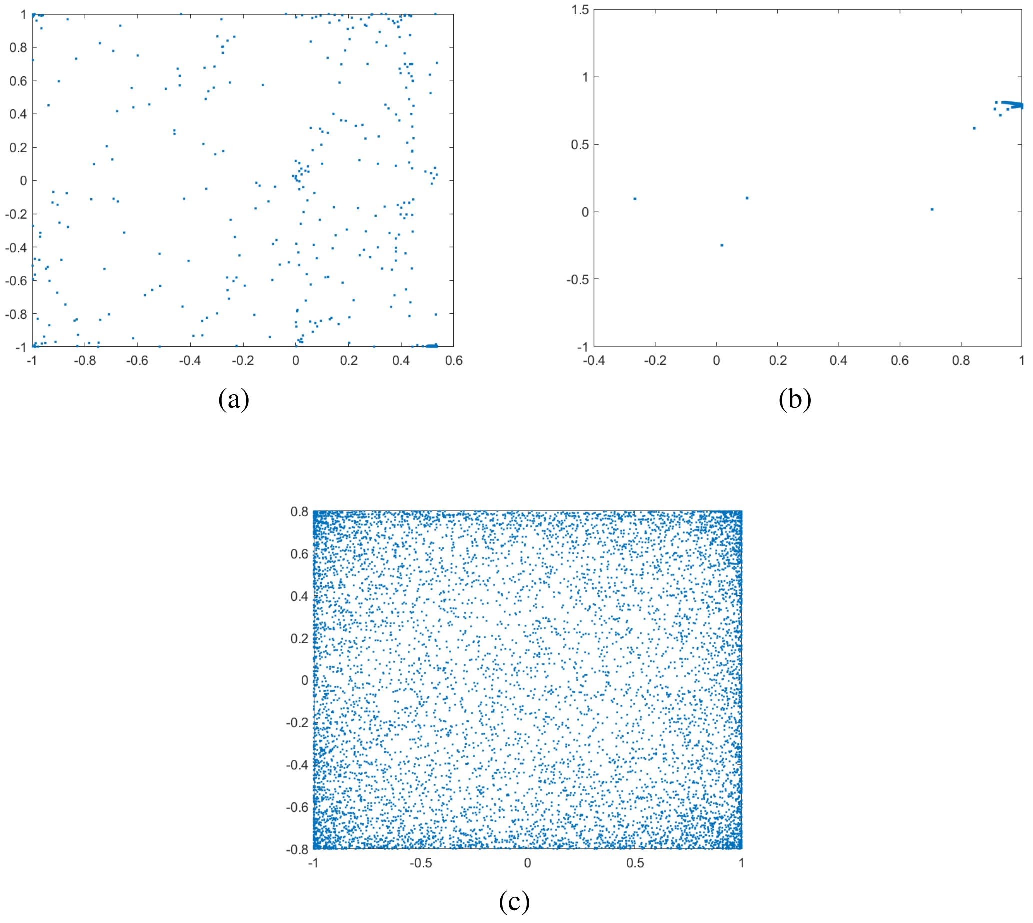


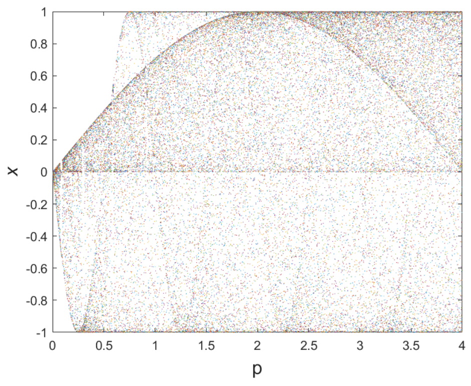
Figure 8.5: Trajectories of 2D chaotic maps: (a) 2D-CLSS, (b) 2D-SCS and (c) 2D-SHCM

8.3.3 Bifurcation

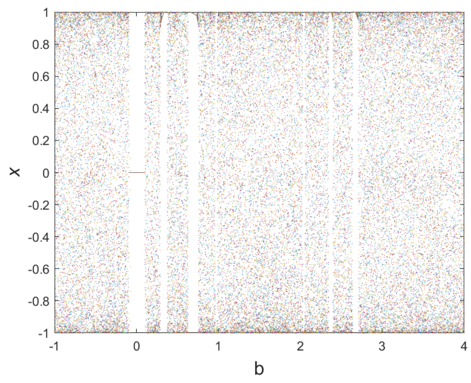
Bifurcation diagram shows the connections between the chaotic points and the control parameters. The dotted area in the bifurcation diagram indicates the chaotic region. Bifurcation diagrams are plotted for the control parameters $p \in (0,4)$ for 2D-CLSS and $a,b \in (-1,4)$ for 2D-SCS and 2D-SHCM. From the Figure 8.6, it demonstrate the 2D-SHCM has the least periodic windows and all the chaotic points are widely distributed over the range of the control parameters.

8.3.4 State mapping network

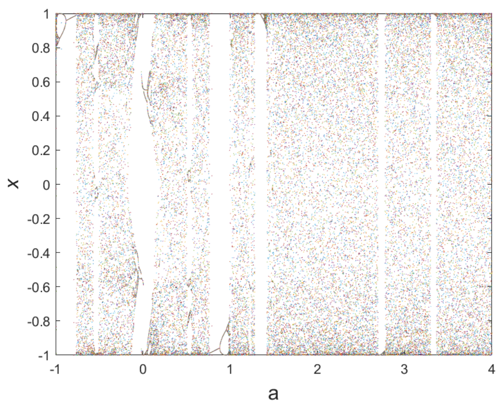
When a chaotic map is generated using a digital device with finite precision, the resulting chaos fails to maintain the dynamic characteristics of the original chaotic map in the continuous domain. To study the the dynamical behaviours of chaotic maps in digital domain, state mapping network (SMN) or functional graph are utilized to observe the iterative trajectories of the digital chaotic maps. We apply the techniques given by Li, Feng, Li, Kurths and Chen (2019); Li et al. (2021) which can be constructed in the following way: the $(2^n)^2$ possible states are are considered as $(2^n)^2$ nodes, where n is the fixed-point arithmetic precision. Every node (x_1,y_1) is connected with a directed edge to their corresponding node (x_2,y_2) by using integer quantization function. In this chapter, we plot the SMN for 2D-SHCM with the control parameters $a = 1.4, b = 0.3$ and $n = 3$, by which the quantized outputs of the chaotic map will fall into 64 possible states. Then, the results are compared to 2D-CLSS with $p = 2$ and 2D-SCS with $a = 2, b = 1.5$. The SMNs are given in Figure 8.7 and comparisons of the period distribution of



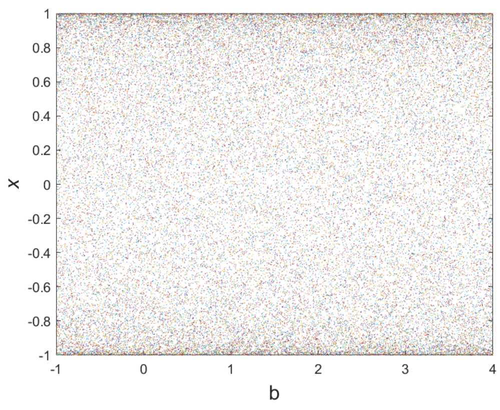
(a)



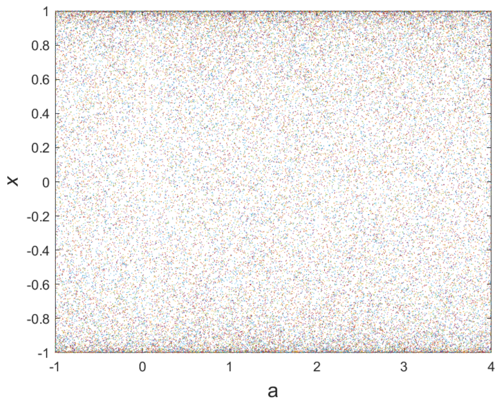
(b)



(c)



(d)



(e)

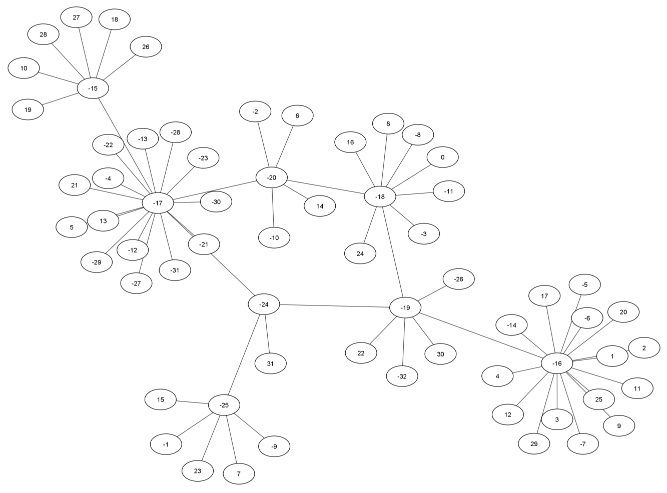
Figure 8.6: Bifurcation diagrams of 2D chaotic maps: (a) 2D-CLSS, (b) 2D-CHS when $b = 0.3$, (c) 2D-CHS when $a = 1.4$ (d) 2D-SHCM when $b = 0.3$ and (e) 2D-SHCM when $a = 1.4$

Table 8.1: Comparisons of period distribution

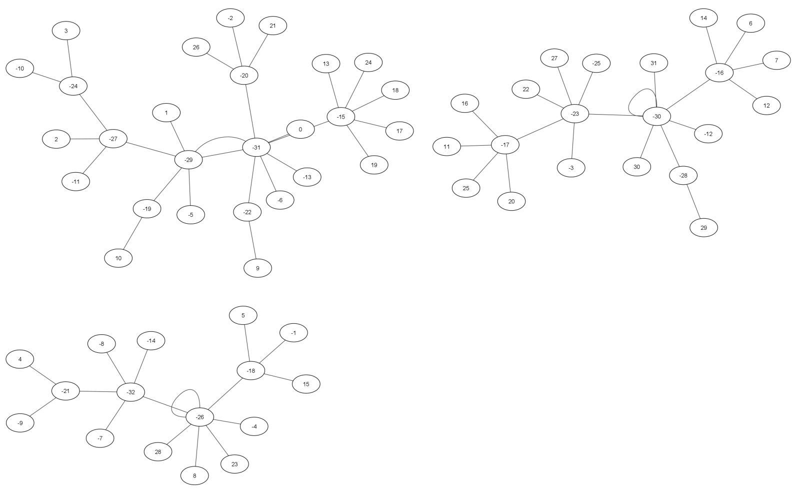
Chaotic map	Max. transient length	Max. cycle length	No. of cycles	No. of fixed points
2D-SHCM (Proposed)	7	2	1	0
2D-CLSS	2	5	1	0
2D-SCS	3	2	3	2

the three chaotic maps are shown in Table 8.1.

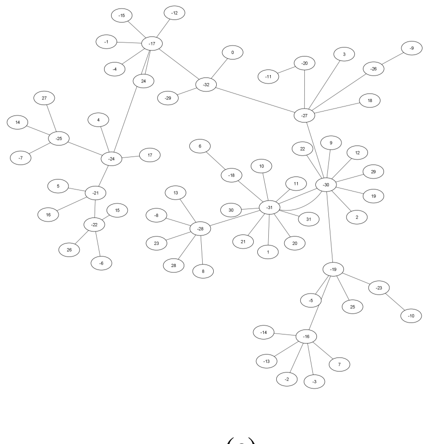
From Figure 8.7, we observed that 2D-SCS has two cycles, while 2D-CLSS and 2D-SHCM have one cycle only. It also observed that the maximal transient length of the proposed chaotic map is 7, where 2D-CLSS and 2D-SCS are 2 and 3, respectively. The proposed map has the longest maximal transient length, meaning that the state variables requires to iterate more times before entering into a cycle. From Figure 8.7b, it is evident that 2D-SCS tends to converge into a fixed point, which is depicted by a self-loop in one of the cycles. This indicates the potential existence of equivalent keys of equivalent keys that can generate the identical chaotic sequences. For 2D-CLSS and 2D-SHCM maps, they enter into a cycle with the cycle length of 5 and 2. The total cycle length is the sum of the maximal transient length and the maximum cycle length. The proposed chaotic map exhibits the longest cycle length compared to the other chaotic maps, demonstrating that our proposed chaotic maps outperform the others.



(a)



(b)



(c)

Figure 8.7: State mapping network of 2D chaotic maps: (a) 2D-CLSS, (b) 2D-SCS and (c) 2D-SHCM

8.4 Proposed image encryption scheme

8.4.1 Notations

Some notations used in this chapter are listed as follows.

- The bold uppercase letters are used to represent an assembly, which can be an array or sequence, a matrix, or a 3D color image. A plain color image **P** with size $3 \times M \times N$ consists of three color planes **R**, **G** and **B** with each color plane consists of $M \times N$ pixels. Each pixel of a color plane ranges from 0 to 255 and consists of 8 bits. Each color plane is divided into eight bitplanes using the BBD technique Zhou, Cao and Chen (2014), i.e.,

  $$\begin{aligned} \mathbf{R} &= \{\mathbf{R_1}, \mathbf{R_2}, \ldots, \mathbf{R_8}\}, \\ \mathbf{G} &= \{\mathbf{G_1}, \mathbf{G_2}, \ldots, \mathbf{G_8}\}, \\ \mathbf{B} &= \{\mathbf{B_1}, \mathbf{B_2}, \ldots, \mathbf{B_8}\}. \end{aligned} \tag{8.7}$$

  The i^{th} bitplane represents the collection of the i^{th} bit of all pixels in the corresponding channel, for $i \in \{1, 2, \ldots, 8\}$. The i^{th} bitplane can be represented in the matrix form as

$$\mathbf{R_i} = \begin{pmatrix} R^i_{1,1} & R^i_{1,2} & \cdots & R^i_{1,N} \\ R^i_{2,1} & R^i_{2,2} & \cdots & R^i_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ R^i_{M,1} & R^i_{M,2} & \cdots & R^i_{M,N} \end{pmatrix}, \mathbf{G_i} = \begin{pmatrix} G^i_{1,1} & G^i_{1,2} & \cdots & G^i_{1,N} \\ G^i_{2,1} & G^i_{2,2} & \cdots & G^i_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ G^i_{M,1} & G^i_{M,2} & \cdots & G^i_{M,N} \end{pmatrix},$$

$$\mathbf{B_i} = \begin{pmatrix} B^i_{1,1} & B^i_{1,2} & \cdots & B^i_{1,N} \\ B^i_{2,1} & B^i_{2,2} & \cdots & B^i_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ B^i_{M,1} & B^i_{M,2} & \cdots & B^i_{M,N} \end{pmatrix}. \tag{8.8}$$

- **LSB** and **MSB** denote the least significant bit block and the most significant bit block, respectively. The **LSB** is composed of the lower four bitplanes of the three color planes, while the **MSB** consists of the higher four bitplanes of the three color planes. Each block size is $12M \times N$. The equation of **LSB** and **MSB** blocks are given by

$$\begin{aligned} \mathbf{LSB} &= \begin{pmatrix} l_{1,1} & l_{1,2} & \cdots & l_{1,N} \\ l_{2,1} & l_{2,2} & \cdots & l_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ l_{12M,1} & l_{12M,2} & \cdots & l_{12M,N} \end{pmatrix}, \\ &= \left(\mathbf{R_1}, \mathbf{G_1}, \mathbf{B_1}, \mathbf{R_2}, \mathbf{G_2}, \mathbf{B_2}, \mathbf{R_3}, \mathbf{G_3}, \mathbf{B_3}, \mathbf{R_4}, \mathbf{G_4}, \mathbf{B_4}\right)^T, \end{aligned} \tag{8.9}$$

$$\begin{aligned} \mathbf{MSB} &= \begin{pmatrix} m_{1,1} & m_{1,2} & \cdots & m_{1,N} \\ m_{2,1} & m_{2,2} & \cdots & m_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ m_{12M,1} & m_{12M,2} & \cdots & m_{12M,N} \end{pmatrix}, \\ &= \left(\mathbf{R_5}, \mathbf{G_5}, \mathbf{B_5}, \mathbf{R_6}, \mathbf{G_6}, \mathbf{B_6}, \mathbf{R_7}, \mathbf{G_7}, \mathbf{B_7}, \mathbf{R_8}, \mathbf{G_8}, \mathbf{B_8}\right)^T, \end{aligned} \tag{8.10}$$

where $\mathbf{R_i}$, $\mathbf{G_i}$ and $\mathbf{B_i}$ are given by Equation (8.8).

- The operation $\oplus$ denotes the bitwise logical exclusive OR (XOR) of two bit strings.
- Let $\mathbf{X}$ be a sequence of length L. The operation $[\mathbf{Z_x}, \mathbf{Idx_x}] = sort(\mathbf{X})$ denotes a new sequence $\mathbf{Z_x}$ obtained by sorting $\mathbf{X}$ in ascending order. Corresponding to the elements in $\mathbf{Z_x}$, their indices in $\mathbf{X}$ will form a new sequence $\mathbf{Idx_x}$, which is a random permutation of the integers within the range 1 to L.
- The operation $floor$ denotes rounding a number down to the nearest integer.

8.4.2 Intrinsic properties of image

The information percentage of every bitplane can be calculated based on Equation (8.11) and the results are shown in Table 8.2. Figure 8.8 visualizes the amount of image information for color plane **B** in eight bitplanes.

$$B(i) = \frac{2^{i-1}}{\sum_{i=1}^{8} 2^{i-1}}, \tag{8.11}$$

where i refers to the i^{th} bitplane. From Table 8.2, it shows that the lower four

Table 8.2: Information percentage of each bitplane of a color plane

Bitplane, i	Percentage (%)	Bitplane, i	Percentage (%)
1	0.39	5	6.27
2	0.78	6	12.55
3	1.57	7	25.10
4	3.14	8	50.20

bitplanes for $i \in \{1,2,3,4\}$ carry less information as they contribute to a total of 5.88% of the image information, while the higher four bitplanes for $i \in \{5,6,7,8\}$ carry most of the visually meaningful data as they cover 94.12%

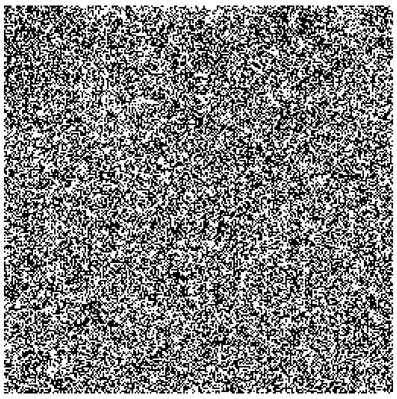
of the image information. The inherent properties of the bit distribution motivates us to randomly shuffle the bits from the higher bitplanes with the bits from lower bitplanes. Another properties is that majority of the bits at the 8^{th} bitplane has the opposite value from the corresponding bits at 7^{th} bitplane. It is demonstrated by Figures 8.8g and 8.8h. The processing of interchanging bits between higher and lower bitplanes can help to eliminate the strong correlation within the higher bitplanes. In our proposed encryption scheme, the shuffling process will be done by the crossover process and then followed by the mutation and non-sequential diffusion process.

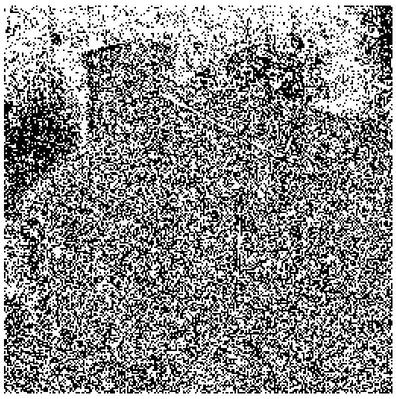
8.4.3 Pseudorandom sequences generation

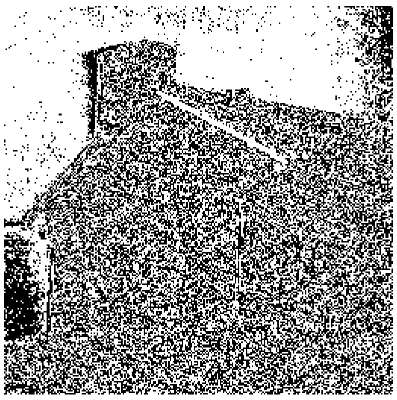
The large key space is important to resist the brute force attack. The secret keys includes the initial conditions $x_0, y_0 \in [-1,1]$ and control parameters $a, b \in \mathbf{R}$ of 2D-SHCM. A secure hash function SHA-256 is applied on the plain image and its digest is used to obtain the secret keys. Using SHA-256 can make a slight change of the plain image to generate a totally different digest, and hence enhance the sensitivity of the image encryption to the plain image. Thirty two 8-bit blocks are generated by the hash function as

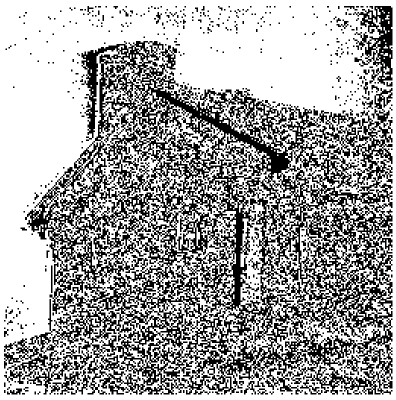
$$\mathbf{H} = \{h_1, h_2, \dots, h_{32}\}. \tag{8.12}$$

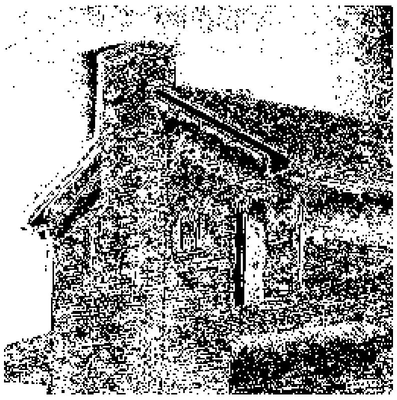
To produce two chaotic sequences $\mathbf{X}$ and $\mathbf{Y}$, the initial conditions and the control parameters of 2D-SHCM in Equation (8.6), x_0, y_0, a and b are updated

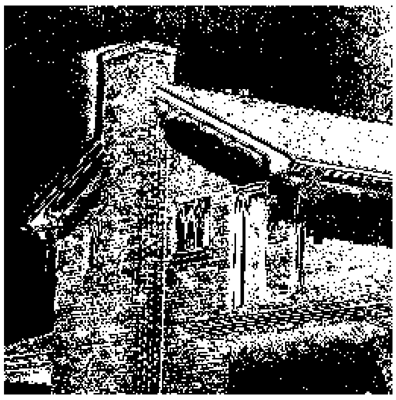

(a) 1st bitplane

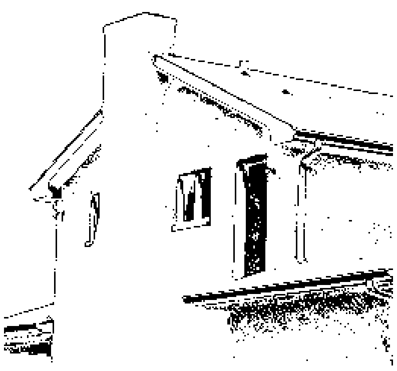

(b) 2nd bitplane


(c) 3rd bitplane


(d) 4th bitplane


(e) 5th bitplane


(f) 6th bitplane


(g) 7th bitplane

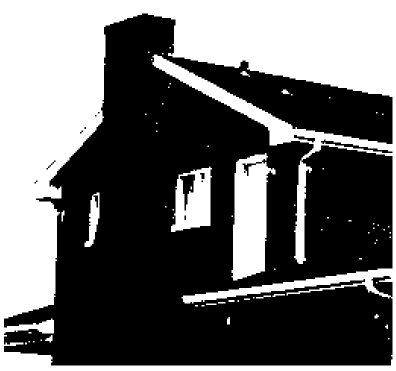

(h) 8th bitplane

Figure 8.8: The eight bitplanes of color plane **B**

using the digest **H** by

$$\begin{cases} x_0' &= x_0 + \frac{h_1 \oplus h_2 \oplus \cdots \oplus h_8}{2^{15}}, \\ y_0' &= y_0 + \frac{h_9 \oplus h_{10} \oplus \cdots \oplus h_{16}}{2^{15}}, \\ a' &= a + \frac{h_{17} \oplus h_{18} \oplus \cdots \oplus h_{24}}{2^{15}}, \\ b' &= b + \frac{h_{25} \oplus h_{26} \oplus \cdots \oplus h_{32}}{2^{15}}. \end{cases} \tag{8.13}$$

We iterate the 2D-SHCM for $T + 15rMN$ times, where the first T elements are discarded to avoid the harmful transient effect, and r is the number of encryption rounds. The resulting chaotic sequences are $\mathbf{X} = \{x_i\}_{i=1}^{15rMN}$ and $\mathbf{Y} = \{y_i\}_{i=1}^{15rMN}$. Then, these two sequences are divided into following sequences.

$$\begin{aligned}
\mathbf{X_1} &= \{x_{1i}\}_{i=1}^{12MN} = \{x_i\}_{i=15(r-1)MN+1}^{[15(r-1)+12]MN}, \\
\mathbf{Y_1} &= \{y_{1i}\}_{i=1}^{12MN} = \{y_i\}_{i=15(r-1)MN+1}^{[15(r-1)+12]MN}, \\
\mathbf{X_2} &= \{x_{2i}\}_{i=1}^{12M} = \{x_i\}_{i=[15(r-1)+12]MN+1}^{[15(r-1)+12]MN+12M}, \\
\mathbf{Y_2} &= \{y_{2i}\}_{i=1}^{12M} = \{y_i\}_{i=[15(r-1)+12]MN+1}^{[15(r-1)+12]MN+12M}, \\
\mathbf{X_3} &= \{x_{3i}\}_{i=1}^{N} = \{x_i\}_{i=[15(r-1)+12]MN+12M+1}^{[15(r-1)+12]MN+12M+N}, \\
\mathbf{Y_3} &= \{y_{3i}\}_{i=1}^{N} = \{y_i\}_{i=[15(r-1)+12]MN+12M+1}^{[15(r-1)+12]MN+12M+N}, \\
\mathbf{X_4} &= \{x_{4i}\}_{i=1}^{M} = \{x_i\}_{i=[15(r-1)+12]MN+12M+N+1}^{[15(r-1)+12]MN+12M+N+M}, \\
\mathbf{Y_4} &= \{y_{4i}\}_{i=1}^{3N} = \{y_i\}_{i=[15(r-1)+12]MN+12M+N+1}^{[15(r-1)+12]MN+12M+4N}, \\
\mathbf{X_5} &= \{x_{5i}\}_{i=1}^{3MN} = \{x_i\}_{i=[15(r-1)+12]MN+1}^{[(r-1)+1]15MN}, \\
\mathbf{Y_5} &= \{y_{5i}\}_{i=1}^{3MN} = \{y_i\}_{i=[15(r-1)+12]MN+1}^{[(r-1)+1]15MN}.
\end{aligned} \tag{8.14}$$

8.4.4 Encryption

The overview of the proposed scheme is graphically presented in Figure 8.9. The detailed process is given as follows:

Input The $3 \times M \times N$ color image **P**, Secret keys x_0, y_0, a and b.

Output Cipher image **C**.

1. Generate psedorandom sequences according to the process described in Subsection 8.4.3.

2. Divide the color image **P** into three color planes: **R**, **G** and **B**. Decompose each color plane into eight bitplanes by using Equations (8.7) and (8.8).

3. Obtain **LSB** and **MSB** according to Equations (8.9) and (8.10), respectively.

4. Perform mutation that has been discussed in Section 3.2.2 as follows:

   (a) Compute $\mathbf{X_1'} = \{x_i'\}_{i=1}^{12MN}$ and $\mathbf{Y_1'} = \{y_i'\}_{i=1}^{12MN}$ from $\mathbf{X_1}$ and $\mathbf{Y_1}$ given in Equation (8.14) as

$$x_i' = \begin{cases} 1, & \text{if } x_{1i} > 0; \\ 0, & \text{otherwise,} \end{cases} \quad y_i' = \begin{cases} 1, & y_{1i} > 0; \\ 0, & \text{otherwise,} \end{cases} \tag{8.15}$$

   Reshape $\mathbf{X_1'}$ and $\mathbf{Y_1'}$ into a $12MN$ blocks as

$$\mathbf{X_1'} = \begin{pmatrix} x'_{1,1} & x'_{1,2} & \cdots & x'_{1,N} \\ x'_{2,1} & x'_{2,2} & \cdots & x'_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ x'_{12M,1} & x'_{12M,2} & \cdots & x'_{12M,N} \end{pmatrix}, \tag{8.16}$$

$$\mathbf{Y_1'} = \begin{pmatrix} y'_{1,1} & y'_{1,2} & \cdots & y'_{1,N} \\ y'_{2,1} & y'_{2,2} & \cdots & y'_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ y'_{12M,1} & y'_{12M,2} & \cdots & y'_{12M,N} \end{pmatrix}, \tag{8.17}$$

   (b) Calculate the hamming weight of every byte in $\mathbf{X_1'}$ and $\mathbf{Y_1'}$ that obtained from Equations (8.16) and (8.17) to obtain $\mathbf{HW1} = \{hw1_{i,j}\}_{i=1,j=1}^{12M,N/8}$ and

$\mathbf{HW2} = \{HW2_{i,j}\}_{i=1,j=1}^{12M,N/8}$ as

$$hw1_{i,j+1} = \sum_{k=8j+1}^{8j+8} x'_{i,k}, \quad hw2_{i,j+1} = \sum_{k=8j+1}^{8j+8} y'_{i,k}. \tag{8.18}$$

for $i = 1, \ldots, 12M$ and $j = 0, 1, \ldots, \frac{N}{8} - 1$.

(c) Based on Equation (8.5), perform mutation on **LSB** from Equation (8.9) and **MSB** from Equation (8.10) to obtain two new blocks $\mathbf{V1} = \{v1_{i,k}\}_{i=1,k=1}^{12M,N}$ and $\mathbf{W1} = \{w1_{i,k}\}_{i=1,k=1}^{12M,N}$ as

$$v1_{i,8j+1:8j+8} = Mutation(l_{i,8j+1:8j+8}, hw1_{i,j+1}), \tag{8.19}$$

$$w1_{i,8j+1:8j+8} = Mutation(m_{i,8j+1:8j+8}, hw2_{i,j+1}), \tag{8.20}$$

for $i = 1, \ldots, 12M$ and $j = 0, 1, \ldots, \frac{N}{8} - 1$.

5. Perform row-wise permutation process using two-point crossover and one-point crossover that are discussed in Subsections 8.2.2 and 8.2.1 as follows:

(a) Using $\mathbf{X_2}$ and $\mathbf{Y_2}$ from Equation (8.14) to obtain index vectors $\mathbf{Idx_{x2}}$ and $\mathbf{Idx_{y2}}$ as

$$[\mathbf{Z_{x2}}, \mathbf{Idx_{x2}}] = \text{sort}(\mathbf{X_2}), \tag{8.21}$$

$$[\mathbf{Z_{y2}}, \mathbf{Idx_{y2}}] = \text{sort}(\mathbf{Y_2}). \tag{8.22}$$

(b) Generate two arrays $\mathbf{F} = \{f_i\}_{i=1}^{12M}$ and $\mathbf{G} = \{g_i\}_{i=1}^{12M}$ using $\mathbf{X_2}$ and $\mathbf{Y_2}$ for the crossover points by

$$f_i = \text{floor}((x_{2i} \times 10^{15}) \bmod N) + 1, \tag{8.23}$$

$$g_i = \text{floor}((y_{2i} \times 10^{15}) \bmod N) + 1, \tag{8.24}$$

where $i = 1, \ldots, 12M$ and N is the column size.

(c) For $i = 1$, let $\mathbf{Idx_{x2}}(i)^{th}$ row of the $\mathbf{V1}$ be the parent bit string A and

$\mathbf{Idx_{y2}}(i)^{th}$ row of the **W1** be the parent bit string B, f_i and g_i be the crossover points. Perform crossover to obtain child bit strings $\big(\mathbf{V2}(i,:),\mathbf{W2}(i,:)\big) = Crossover\big(\mathbf{V1}(\mathbf{Idx_{x2}}(i),:),\mathbf{W1}(\mathbf{Idx_{y2}}(i),:),f_i,g_i\big)$, where *Crossover* can be either $Crossover_1$ or $Crossover_2$ depending on Equations (8.1), (8.2) and (8.3).

(d) Repeat Step 5c for $i = 2,\ldots 12M$ to obtain two blocks **V2** and **W2**.

6. Perform column-wise permutation process using uniform crossover that are discussed in Subsection 8.2.3 as follows:

(a) Using $\mathbf{X_3}$ and $\mathbf{Y_3}$ from Equation (8.14) to obtain index vectors $\mathbf{Idx_{x3}}$ and $\mathbf{Idx_{y3}}$ as

$$[\mathbf{Z_{x3}},\mathbf{Idx_{x3}}] = \text{sort}(\mathbf{X_3}), \tag{8.25}$$

$$[\mathbf{Z_{y3}},\mathbf{Idx_{y3}}] = \text{sort}(\mathbf{Y_3}). \tag{8.26}$$

(b) For $i = 1$, let $\mathbf{Idx_{x3}}(i)^{th}$ column of the **V2** be the parent bit string A and $\mathbf{Idx_{y3}}(i)^{th}$ row of the **W2** be the parent bit string B.

(c) To determine the crossover region, select $\mathbf{Idx_{x3}}(i)^{th}$ and $\mathbf{Idx_{y3}}(i)^{th}$ columns of the respective $\mathbf{X_1'}$ and $\mathbf{Y_1'}$ from Equations (8.16) and (8.17) be the reference arrays C and D. Perform uniform crossover based on Equation (8.4) to obtain child bit strings $\big(\mathbf{V3}(:,i),\mathbf{W3}(:,i)\big) = Crossover_u\big(\mathbf{V2}(:,\mathbf{Idx_{x3}}(i)),\mathbf{W2}(:,\mathbf{Idx_{y3}}(i)),\mathbf{X_1'}(:,\mathbf{Idx_{x3}}(i)),$
$\mathbf{Y_1'}(:,\mathbf{Idx_{y3}}(i))\big)$.

(d) Repeat Steps 6b and 6c for $i = 2,\ldots N$ to obtain two blocks **V3** and **W3**.

7. Form a block $\mathbf{U} = \{u_{i,j}\}_{i=1,j=1}^{M,3N}$ with $M \times 3N$ pixels by combining **V3** and **W3**, where pixels from column 1 until N are **R** color plane, column $N+1$ until $2N$ are **G** color plane, and column $2N+1$ until $3N$ are **B** color plane.

8. Apply non-linear diffusion as follows:

(a) Using $\mathbf{X_4}$ and $\mathbf{Y_4}$ from Equation (8.14) to obtain index vectors $\mathbf{Idx_{x4}}$ and $\mathbf{Idx_{y4}}$ as

$$[\mathbf{Z_{x4}}, \mathbf{Idx_{x4}}] = \text{sort}(\mathbf{X_4}), \tag{8.27}$$

$$[\mathbf{Z_{y4}}, \mathbf{Idx_{y4}}] = \text{sort}(\mathbf{Y_4}). \tag{8.28}$$

(b) Obtain two sequences $\mathbf{X'_5} = \{x''_i\}_{i=1}^{3MN}$ and $\mathbf{Y'_5} = \{y''_i\}_{i=1}^{3MN}$ from $\mathbf{X_5}$ and $\mathbf{Y_5}$ from Equation (8.14) using quantization as

$$x''_i = \text{floor}((x_{5i} \times 10^{15}) \bmod 256), \tag{8.29}$$

$$y''_i = \text{floor}((y_{5i} \times 10^{15}) \bmod 256). \tag{8.30}$$

(c) Let $\mathbf{U'}\big(\mathbf{Idx_{x4}}(1), \mathbf{Idx_{y4}}(0)\big) = \sum_{i=1}^{M} \sum_{j=1}^{3N} u_{i,j} \bmod 256$. Compute

$$\mathbf{U'}\big(\mathbf{Idx_{x4}}(i), \mathbf{Idx_{y4}}(j)\big) = \begin{cases} \big[(x''_k + y''_k) \bmod 256\big] \oplus \\ \quad \Big[\big(x''_k + \big(\mathbf{U'}\big(\mathbf{Idx_{x4}}(i-1), \mathbf{Idx_{y4}}(3N)\big) \\ \quad \oplus \mathbf{U}\big(\mathbf{Idx_{x4}}(i), \mathbf{Idx_{y4}}(j)\big)\big)\big) \bmod 256\Big], \\ \quad \text{for } i > 1, j = 1, \\ \big[(x''_k + y''_k) \bmod 256\big] \oplus \\ \quad \Big[\big(x''_k + \big(\mathbf{U'}\big(\mathbf{Idx_{x4}}(i), \mathbf{Idx_{y4}}(j-1)\big) \\ \quad \oplus \mathbf{U}\big(\mathbf{Idx_{x4}}(i), \mathbf{Idx_{y4}}(j)\big)\big)\big) \bmod 256\Big], \\ \quad \text{otherwise,} \end{cases} \tag{8.31}$$

where $i = 1, \ldots, M$, $j = 1, \ldots, 3N$ and $k = 1, \ldots, 3MN$.

9. Arrange $\mathbf{U'}$ into **R, G, B** channels. Repeat Step 2 until Step 8 for remaining $r-1$ rounds to obtain the cipher image $\mathbf{C}$.

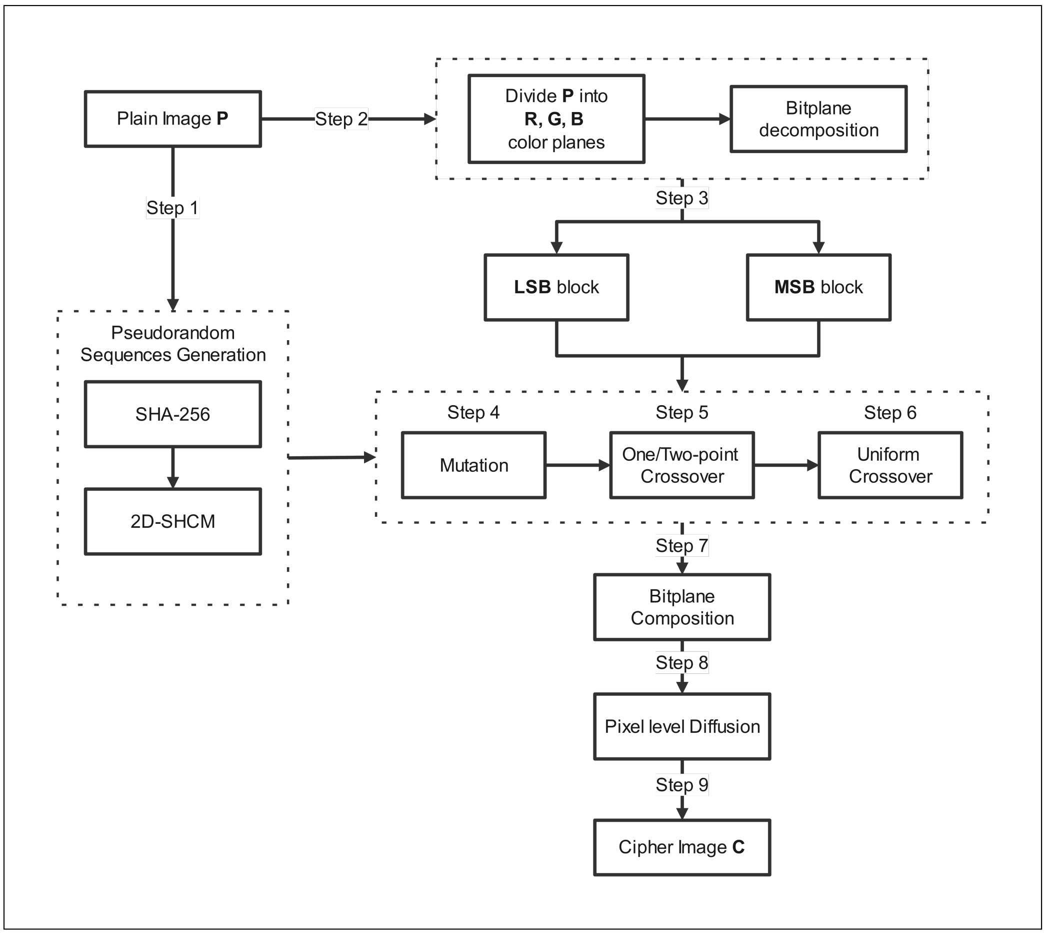


Figure 8.9: Encryption Algorithm

8.4.5 Decryption

The decryption is the reverse process of encryption as follows:

Input The cipher image **C**, Secret keys x_0, y_0, a and b.

Output Recovered image **P**.

1. Repeat Step 1.
2. Divide cipher image into **R, G** and **B** color planes. Decompose each color plane into eight bitplanes.
3. Perform the inverse of diffusion process to obtain block **U** as follows:

(a) Obtain $\mathbf{Idx_{x4}}$,$\mathbf{Idx_{y4}}$, and $\mathbf{X}'_5$ and $\mathbf{Y}'_5$ based on Equations (8.27)-(8.30).

(b) Obtain block $\mathbf{U}$ by computing

$$\mathbf{U}\big(\mathbf{Idx_{x4}}(i),\mathbf{Idx_{y4}}(j)\big)=\begin{cases}\Big[\big(\mathbf{U}'\big(\mathbf{Idx_{x4}}(i),\mathbf{Idx_{y4}}(j)\big)\oplus\\ \quad(x''_k+y''_k)-x''_k\big) \bmod 256\Big]\oplus\\ \quad\mathbf{U}'\big(\mathbf{Idx_{x4}}(i-1),\mathbf{Idx_{y4}}(3N)\big),\\ \quad\text{for } i>1, j=1,\\ \Big[\big(\mathbf{U}'\big(\mathbf{Idx_{x4}}(i),\mathbf{Idx_{y4}}(j)\big)\oplus\\ \quad(x''_k+y''_k)-x''_k\big) \bmod 256\Big]\oplus\\ \quad\mathbf{U}'\big(\mathbf{Idx_{x4}}(i),\mathbf{Idx_{y4}}(j-1)\big),\\ \quad\text{for } \forall i, j>1,\end{cases}\tag{8.32}$$

where $i=1,\dots,M$, $j=1,\dots,3N$ and $k=1,\dots,3MN$. Extract $\mathbf{U}\big(\mathbf{Idx_{x4}}(1),\mathbf{Idx_{y4}}(1)\big)$ that satisfying the following equation.

$$\begin{aligned}\mathbf{U}\big(\mathbf{Idx_{x4}}(1),\mathbf{Idx_{y4}}(1)\big)=&\big[\big(\mathbf{U}'\big(\mathbf{Idx_{x4}}(1),\mathbf{Idx_{y4}}(1)\big)\oplus\\ &(x''_k+y''_k)-x''_k\big) \bmod 256\big]\\ &\oplus\Bigg(\sum_{\substack{i=1\\ i\neq1\&\&j\neq1}}^{M}\sum_{j=1}^{3N}\mathbf{U}\big(\mathbf{Idx_{x4}}(i),\mathbf{Idx_{y4}}(j)\big)+\\ &\mathbf{U}\big(\mathbf{Idx_{x4}}(1),\mathbf{Idx_{y4}}(1)\big)\Bigg) \bmod 256.\end{aligned}\tag{8.33}$$

4. Split block $\mathbf{U}$ into $\mathbf{V3}$ and $\mathbf{W3}$.

5. Perform the inverse of uniform crossover process as follows:

(a) Obtain $\mathbf{Idx_{x3}}$ and $\mathbf{Idx_{y3}}$ using Equations (8.25) and (8.26).

(b) Obtain $\mathbf{X}'_1$ and $\mathbf{Y}'_2$ by using Equations (8.15)-(8.17).

(c) Obtain $\mathbf{V2}$ and $\mathbf{W2}$ by reversing the uniform crossover process in Step 6, i.e., $\big(\mathbf{V2}(:,\mathbf{Idx_{x3}}(i)),\mathbf{W2}(:,\mathbf{Idx_{y3}}(i))\big) = \mathit{Crossover}_u\big(\mathbf{V3}(:,i),\mathbf{W3}(:$

$,i),\mathbf{X_1'}(:,\mathbf{Idx_{x3}}(i)),\mathbf{Y_1'}(:,\mathbf{Idx_{y3}}(i))\big)$ for $i=1,\dots N$.

6. Perform the inverse of two-point or one-point crossover process as follows:

   (a) Obtain $\mathbf{Idx_{x2}}$, $\mathbf{Idx_{y2}}$, **F** and **G** using Equations (8.21)-(8.24).

   (b) Obtain **V1** and **W1** by reversing the crossover process in Step 5, i.e., $\big(\mathbf{V1}(\mathbf{Idx_{x2}}(i),:),\mathbf{W1}(\mathbf{Idx_{y2}}(i),:)\big)=Crossover\big(\mathbf{V2}(i,:),\mathbf{W2}(i,:),f_i,g_i\big)$

7. Perform the inverse of mutation process as follows.

   (a) Obtain the hamming weight **HW1** and **HW2** using Equation (8.18).

   (b) Obtain **LSB** and **MSB** by reversing the mutation process in Step 4.

$$l_{i,8j+1:8j+8}=Mutation(v1_{i,8j+1:8j+8},hw1_{i,j+1}), \tag{8.34}$$

$$m_{i,8j+1:8j+8}=Mutation(w1_{i,8j+1:8j+8},hw2_{i,j+1}), \tag{8.35}$$

8. Arrange **LSB** and **MSB** into **R, G , B** channels and combine them as recovered image **P**.

8.4.6 Discussion

The proposed encryption scheme exhibits the following advantages.

1. The image encryption scheme fulfills the confusion and diffusion properties that required by the secure cryptosystem (Shannon, 1949). The mutation process alters the bit values of the image data and the changes are spread over whole planes through the crossover process. The non-sequential permutation and diffusion process can spread the changes from one bit to another bit in a random order. The bit-level permutation and diffusion shuffle the eight bit planes thoroughly by considering the intrinsic features of the image.

2. The scheme was ended up with a non-linear pixel-level diffusion to further diffuse the three color planes completely. The diffusion effect depends on the keystream elements, pixels from the genetic algorithms and previous encrypted pixel at the random position.

3. The image encryption has the ability to withstand different types of attacks. This is because the encrypted image will be totally different with a small change in the pixel, even with the same secret keys being used. The non-linear diffusion process could magnify the difference in the output of genetic algorithms through the bitwise exclusively or and modular addition operations. If any bit is altered, it will alter the $\mathbf{U}'\left(\mathbf{Idx_{x4}}(1), \mathbf{Idx_{y4}}(0)\right)$ in Step 8c of the encryption process and cause all the bits to be changed after one encryption round. Therefore, the encryption requires one round to obtain good diffusion effect. Its strength of resistance to different types of attacks are also shown in Section 8.5 Security Analysis.

4. The structure of the encryption scheme is simple and requires low computational cost. It requires one encryption round to achieve good security.

8.5 Security Analysis

This section presents the results of various experiments designed to test the security level of image encryption. All tests were conducted using the Matlab R2019a environment on a computer equipped with an Intel® Core™ i5-8250 CPU @ 1.60GHz, 8 GB memory and the Windows 10 operating system. The experiments were performed using a colour image titled “House” from the USC-SIPI image database, which has a size of 256×256. The secret keys employed in the tests are $x_0 = 0.1, y = 0.1, a = 1.4, b = 0.3$.

8.5.1 Key space

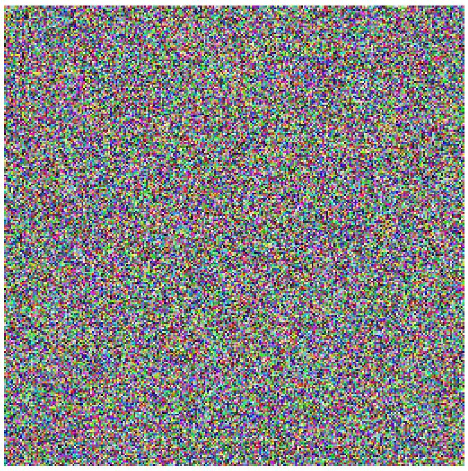
For a secure image encryption, the key space must be large enough to withstand the brute force attack. The secret keys used in the proposed image scheme are $\langle x_0, y_0, a, b \rangle$. Since 2D-SHCM has chaotic behavior when $a, b \in \mathbf{R}$, if the computation precision of floating-point number is around 10^{-15}, then the key space is $(10^{15})^4 \approx 2^{199.32}$, which is large enough to resist the brute–force attack.

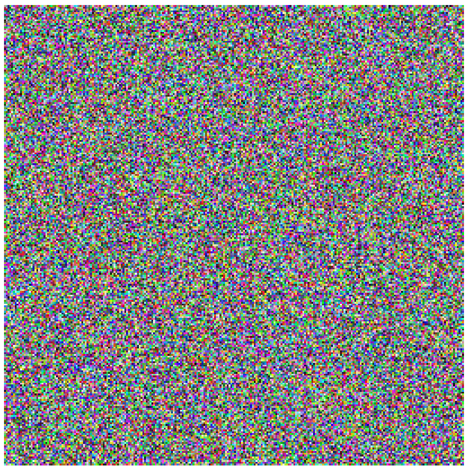
8.5.2 Key sensitivity analysis

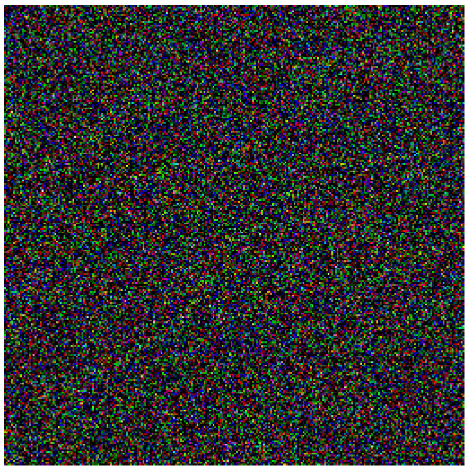
This test measures the sensitivity of cipher image to a tiny changes in secret key. The proposed image encryption is extremely sensitive to its initial condition and control parameters. The initial set of secret keys key_1 ($x_0 = 0.1$ and $y_0 = 0.1$) is changed slightly to key_2 ($x'_0 = 0.1 + 10^{15}$ and $y'_0 = 0.1 + 10^{15}$). Both key_1 and key_2 are used to encrypt the same plain image (refer to Figure 8.10a) and their cipher images are shown in Figures 8.10b and 8.10c, respectively. Figure 8.10d shows the difference in the pixel values of these two images and their difference ratio is 99.6048%, which indicates that 99.6048% of their pixels are different. Thus, a slight change in the secret keys can produce the cipher image that is almost totally different.

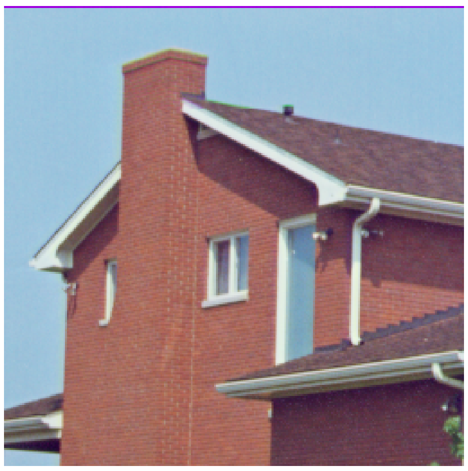
Next, the cipher image in Figure 8.10b is then decrypted by using both key_1 and key_2. Based on the results shown in Figures 8.10e and 8.10f, only key_1 can successfully decrypt the cipher image, while the image decrypted by using key_2 is unreadable.

(a)


(b)


(c)


(d)


(e)

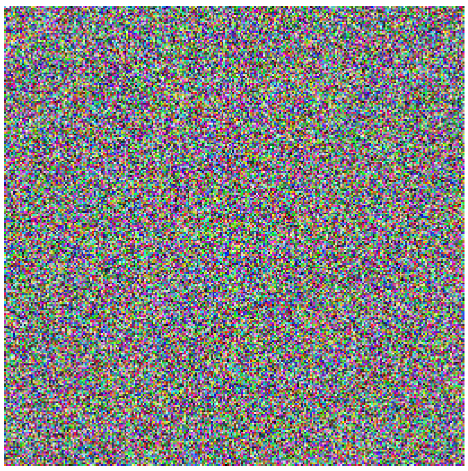

(f)

Figure 8.10: Key Sensitivity Analysis. (a) Plain image. (b) Cipher Image using key_1. (c) Cipher Image using key_2. (d) Difference between Figures 8.10b and 8.10c (e) Recovered image using key_1. (f) Recovered image using key_2.

8.5.3 Resistance to chosen plaintext attack

The chosen plaintext attack is one of the most threatening attack models commonly used by adversaries. In this scenario, adversaries possess the capability to select plaintext images and observe their associated cipher images (Bleichenbacher, 1998). By exploiting the relationship between the chosen plain images and corresponding cipher images, the adversaries can recover the plain image from the cipher image without the need for the secret key.

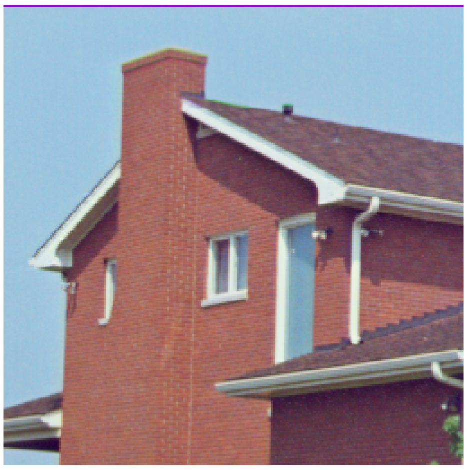
To resist this plaintext attack, our proposed scheme possesses the following features: (a) Chaotic random sequence are generated using the plain image pixel. Small change in the pixel value could result in a totally different sequence. (b) Crossover and mutation of the genetic algorithm introduces the permutation-diffusion properties to the encryption model. The possible of the pixels are shuffled and the slight changes in the plain image could spread over all the pixels of cipher image. (c) The non-linear diffusion process using the sum of pixels values as one of the input also amplify the diffusion effect.

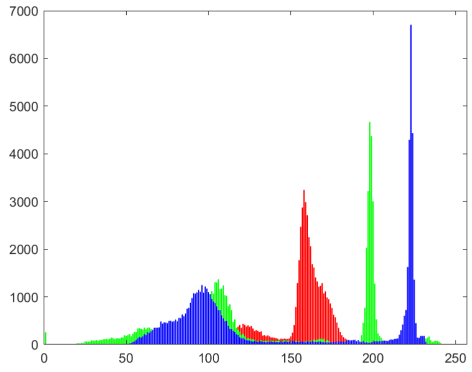
To visually demonstrate the robust capabilities of our proposed scheme in defending against this attack, we conduct the analysis on a color image two special images with pixels are all zero or 255. The tests that we conducted are histogram, the coefficients of correlation, NPCR and UACI and information entropies of cipher images for these two special images.

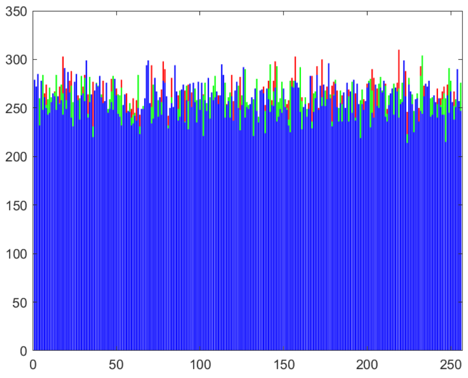
Histogram

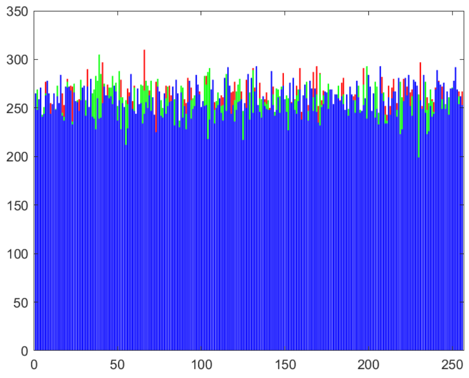
A secure image encryption must ensure that the cipher image is uniformly distributed. Histogram is plotted to show the distribution of the pixel intensity of the image. It is obvious that the pixel values of the encrypted image in Figure 8.11c is fairly uniform and totally different from the histogram of plain image in

Figure 8.11b. The histogram of cipher images for two special images, i.e., all zeros and 255 pixels are plotted in Figures 8.11d and 8.11d. It is observed that the outputs are uniformly distributed.


(a) Plain image "House"


(b) Histogram of plain image


(c) Histogram of encrypted image


(d) Histogram of encrypted all black image

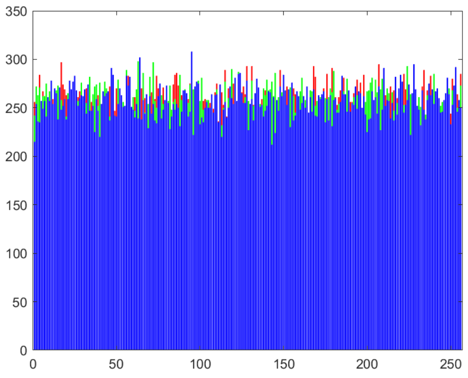

(e) Histogram of encrypted all white image

Figure 8.11: Histogram Analysis

To further verify the uniformity of the pixel distribution, χ^2 test are carried out using

$$\chi^2 = \sum_{i=1}^{k} \frac{(u_i - np)^2}{np}, \tag{8.36}$$

where u_i denotes the frequency of observations at the i^{th} interval and n is the total frequency and $p = \frac{1}{k}$. The smaller the χ^2 value, the closer the distribution of encrypted images to the uniform distribution. As shown in Table 8.3, the proposed encryption scheme passed the test and has a lower χ^2 values for the encrypted image.

Table 8.3: Comparison of χ^2 values of plain and encrypted images for "House" and encrypted special images

Image	χ^2			
	Red	Green	Blue	Average
Plain House	258576.8750	299158.6406	394038.9453	317258.1536
Encrypted House	306.0156	255.7343	298.6718	286.8073
Encrypted all black image	242.0313	232.7656	260.7344	245.1771
Encrypted all white image	280.7109	237.9375	257.0547	258.5677

Correlation Analysis

Images have a high correlation to the adjacent pixels in different directions i.e. horizontal, vertical and diagonal directions. Adversaries could exploit this feature to retrieve the information of the images. To test correlation between the adjacent pixels of the encrypted image, coefficient of adjacent pixels ρ_{xy} is calculated using

$$\rho_{xy} = \frac{\sum_i^N (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{(\sum_{i=1}^N (x_i - \bar{x})^2)(\sum_{i=1}^N (y_i - \bar{y})^2)}}, \tag{8.37}$$

where $\bar{x} = \frac{1}{N}\sum_{i=1}^N x_i$ and $\bar{y} = \frac{1}{N}\sum_{i=1}^N y_i$. From Table 8.4, it can be seen that values obtained by our scheme are much closer to zero, which means that the pixel values are not correlated to each other.

Table 8.4: Correlation coefficient of encrypted images for “House”

Image	Correlation Coefficient		
	Horizontal	**Vertical**	**Diagonal**
Proposed	**-0.00227**	**-0.00036**	**0.00179**
Black	-0.0025	-0.0075	0.0041
White	-0.0025	0.0025	0.0005
Paper Liu and Liu (2020)	-0.0119	-0.0087	-0.0045
Paper Hu et al. (2020)	0.0012	0.0034	0.0017
Paper Alexan et al. (2023)	0.0014	-0.0015	0.0079

NPCR and UACI

Differential attack is an attack exploiting the differences of the chosen pairs of inputs and outputs of a cipher are exploited by the attack (Biham and Shamir, 1991). The commonly used statistical tests to measure the strength of the underlying encryption scheme against differential attack are number of pixel change rate (NPCR) and unified average change intensity (UACI). These tests are applied to two encrypted images C_1 and C_2 with one pixel difference. They are represented by

$$NPCR(C_1,C_2) = \frac{\sum_{i,j} F(i,j)}{N \times M} \times 100\% \tag{8.38}$$

and

$$UACI(C_1 - C_2) = \frac{\sum_{i,j} |C_1(i,j) - C_2(i,j)|}{L \times M \times N} \times 100\%, \tag{8.39}$$

where

$$F(i,j) = \begin{cases} 0 & \text{if } C_1(i,j) = C_2(i,j), \\ 1 & \text{if } C_1(i,j) \neq C_2(i,j), \end{cases} \tag{8.40}$$

and L is the largest allowable pixel value in the image. The ideal expectation values for NPCR and UACI are 99.6094% and 33.4635%, respectively. From Table **??**, it shows that all the test images obtain the results that are higher than the criterion. We can say that our proposed scheme can effectively withstand the differential attack.

Table 8.5: NPCR and UACI scores of encrypted images for “House”

Image	Average NPCR (%)	Average UACI (%)
Proposed	**99.6134**	**33.5323**
Black	99.61700	33.46826
White	99.60175	33.40612
Paper Liu and Liu (2020)	99.6100	32.2000
Paper Hu et al. (2020)	99.6236	33.3619
Paper Alexan et al. (2023)	99.6254	30.5681

Information Entropy

Information entropy serves as a metric for evaluating both the randomness and the distribution of pixels within an image. It is measured by

$$H(m) = -\sum_{i=1}^{L} P[m(i)] \log_2 P[m(i)], \tag{8.41}$$

where $P[m(i)]$ is the probability of $m(i)$, L is the number of pixel values. A higher value of information entropy indicates a more uniform distribution of pixels in the image. The image reaches the theoretical maximum information entropy when each possible pixel value has an equal probability. In other words, the information entropy for an 8-bit image is $H(m)_{max} = \log_2 2^8 = 8$. In this experiment, the information entropy of the cipher image and the special images are closer to 8, indicating satisfactory randomness.

Table 8.6: Information entropy of encrypted images for “House”

Image	Average Information Entropy
Proposed	**7.9989**
Black	7.9974
White	7.9972
Paper Liu and Liu (2020)	7.9897
Paper Hu et al. (2020)	7.9941
Paper Alexan et al. (2023)	7.9967

8.6 Summary

This chapter presents a new image encryption scheme based on 2D-SHCM, the proposed two-dimensional chaotic map, genetic algorithms and SHA-256 hash function. The 2D-SHCM has a better chaotic behavior over the lower-dimensional chaotic system in terms of ergodicity, sensitivity to the initial condition and control parameters, randomness and structural complexity. Furthermore, the use of SHA-256 hash function in modifying the initial conditions of the chaotic map highly improves the sensitivity of the cipher to the change of plain image. The genetic algorithms and the nonlinear diffusion process also introduces a good avalanche effect in the encryption scheme. To test the security level of the proposed scheme, a series of experiments have been conducted. All the numerical results demonstrate that our proposed scheme has good security performance and thus it is suitable for image encryption.

CHAPTER 9

CONCLUSION AND FUTURE WORK

In this chapter, we summarize the key contributions and findings of this thesis. We then conclude with recommendations for future research directions.

9.1 Conclusion

This thesis presents several contributions to the field of chaotic-based image encryption. Firstly, cryptanalysis of existing chaotic-based image encryption schemes was conducted to evaluate their security. A known plaintext attack was applied to the scheme proposed by Biswas et al. (2015), revealing weaknesses in its genetic algorithm-based design. The study also explored the general properties of genetic algorithms and demonstrated how similar attacks could be extended to other encryption systems utilizing genetic algorithms. Additionally, a chosen plaintext attack was performed on the scheme proposed by Ping et al. (2018), analyzing the security and efficiency of the two-point diffusion strategy and highlighting the dynamical degradation of the Henon map. The findings emphasized the need for improved diffusion mechanisms and robust chaotic sources, leading to the proposal of enhanced encryption techniques.

Next, novel chaotic maps were introduced to address the limitations of existing chaotic systems used in encryption. A cascading technique was employed to construct a one-dimensional chaotic map, the Logistic-Beta map, combining the classical logistic map and beta map to improve chaotic behavior. Further, a chaotification approach was developed, integrating a two-dimensional chaotic map with a one-dimensional chaotic map through modular operations. This led to the introduction of the 2D-HenonLog map, which merges the Henon and logistic maps, and the 2D-SHCM map, which combines sine and Henon maps. These maps exhibited improved chaotic properties, such as increased sensitivity to initial conditions, reduced periodic windows, and an extended chaotic range, making them well-suited for pseudorandom number generation and image encryption. A mathematical analysis of their graph structure over a digital device was conducted using state-mapping network analysis, evaluating their dynamical behavior and confirming their suitability for cryptographic applications.

A crucial aspect of this research was the mathematical analysis of the graph structure of chaotic maps in digital devices. In digital implementations, finite precision effects can lead to dynamic degradation, resulting in periodicity, state collisions, and reduced entropy, which ultimately compromise the security of encryption systems. To address these issues, we conducted a state-mapping network analysis comparing the dynamical performance of our proposed chaotic maps with several existing chaotic maps, including the Henon map, improved Henon map, 2D-CLSS, and 2D-SCS. This analysis used a directed graph to illustrate the evolution of digital states, revealing insights into chaotic trajectories under finite precision. The study uncovered that some maps exhibit structural weaknesses, such as attractors with short cycles or biased state transitions, making them vulnerable to cryptanalytic attacks.

Lastly, new chaotic-based image encryption schemes were proposed to enhance security and efficiency. A grayscale image encryption system was designed using a four-dimensional hyperchaotic system with a permutation-diffusion architecture. To further strengthen security, SHA-2 was incorporated to improve sensitivity to plaintext changes. Additionally, a novel color image encryption scheme was introduced based on a two-dimensional Sine-Henon chaotic map and genetic algorithms. This scheme utilized cross-plane permutation, nonlinear diffusion, and dynamic genetic operations, including uniform crossover, to optimize confusion and diffusion. Experimental results demonstrated the robustness of the proposed schemes, showing strong resistance to statistical and differential attacks.

Beyond the direct contributions, this research also has practical applications in secure image transmission and real-time multimedia encryption. The potential for multiple image encryption and parallel image encryption techniques is further explored in Section 9.2 Future Work, providing directions for optimizing encryption efficiency and computational performance. By addressing fundamental weaknesses in existing chaotic encryption schemes and proposing innovative solutions, this thesis contributes to the advancement of secure image encryption methodologies in the digital age.

9.2 Future Work

Based on the research conducted in this thesis, we propose some possible new directions in the field of image encryption. These directions build on the foundation laid by chaotic based encryption and create new opportunities for future improvements in the field. While this research primarily emphasizes the security aspects of encryption, it does not fully address the computational speed of image encryption. Therefore, exploring methods to enhance computational efficiency in future studies would be highly beneficial.

We recommend focusing on multiple image encryption and parallel image encryption, as these methods offer new ways to improve both security and processing speed when handling large volumes of image data.

- **Multiple Image Encryption**: This technique encrypts several images simultaneously using one encryption algorithm. It makes sure that each image maintains confidentiality, integrity, and authenticity under one cryptographic framework. It is particularly useful for secure transmission or storage of multiple images, as it streamlines the encryption process without compromising security. Future research could explore the development of more efficient schemes to handle large volumes of images, addressing the increasing need for bulk data encryption in real-world applications.

- **Parallel Image Encryption**: This technique uses parallel processing to encrypt images concurrently, leveraging the power of multi-core processors or distributed computing environments. It accelerates the encryption process, especially for large datasets or real-time applications. Future research could focus on enhancing the computational efficiency of the parallel encryption algorithms that can processing either segments of a single image or multiple images simultaneously. This is also useful for the applications requiring fast, secure encryption of large volumes of image data.

LIST OF REFERENCES

Abdullah, A. H., Enayatifar, R. and Lee, M., 2012. A hybrid genetic algorithm and chaotic function model for image encryption. *AEU-International Journal of Electronics and Communications*, 66(10), pp. 806–816.

Akkasaligar, P. T. and Biradar, S., 2020. Selective medical image encryption using dna cryptography. *Information Security Journal: A Global Perspective*, 29(2), pp. 91–101.

Alanazi, A. S., Munir, N., Khan, M., Asif, M. and Hussain, I., 2021. Cryptanalysis of novel image encryption scheme based on multiple chaotic substitution boxes. *IEEE Access*, 9, pp. 93795–93802.

Alawida, M., Samsudin, A., Teh, J. S. et al., 2019. Digital cosine chaotic map for cryptographic applications. *IEEE Access*, 7, pp. 150609–150622.

Alexan, W., Elkandoz, M., Mashaly, M., Azab, E. and Aboshousha, A., 2023. Color image encryption through chaos and kaa map. *IEEE Access*, 11, pp. 11541–11554.

Alvarez, G. and Li, S., 2006. Some basic cryptographic requirements for chaos-based cryptosystems. *International journal of bifurcation and chaos*, 16(08), pp. 2129–2151.

Arroyo, D., Amigó Garcia, J. M., Li, S. and Alvarez, G., 2010. *On the inadequacy of unimodal maps for cryptographic applications*.

Bassham III, L. E., Rukhin, A. L., Soto, J., Nechvatal, J. R., Smid, M. E., Barker, E. B., Leigh, S. D., Levenson, M., Vangel, M., Banks, D. L. et al., 2010. Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications.

Behnia, S., Akhshani, A., Mahmodi, H. and Akhavan, A., 2008. Chaotic cryptographic scheme based on composition maps. *International Journal of Bifurcation and chaos*, 18(01), pp. 251–261.

Biham, E., Biryukov, A. and Shamir, A., 1999. Miss in the middle attacks on idea and khufu. *Fast Software Encryption: 6th International Workshop, FSE'99 Rome, Italy, March 24–26, 1999 Proceedings 6*. Springer, pp. 124–138.

Biham, E. and Kocher, P. C., 1994. A known plaintext attack on the PKZIP stream cipher. *International Workshop on Fast Software Encryption*. Springer, pp. 144–153.

Biham, E. and Shamir, A., 1991. Differential cryptanalysis of des-like cryptosystems. *Journal of CRYPTOLOGY*, 4, pp. 3–72.

Biswas, K., Muthukkumarasamy, V. and Singh, K., 2015. An encryption scheme using chaotic map and genetic operations for wireless sensor networks. *IEEE Sensors Journal*, 15(5), pp. 2801–2809.

Bleichenbacher, D., 1998. Chosen ciphertext attacks against protocols based on the rsa encryption standard pkcs# 1. *Annual International Cryptology Conference*. Springer, pp. 1–12.

Boriga, R., Dăscălescu, A. C. and Priescu, I., 2014. A new hyperchaotic map and its application in an image encryption scheme. *Signal Processing: Image Communication*, 29(8), pp. 887–901.

Bouteghrine, B., Tanougast, C. and Sadoudi, S., 2021. Novel image encryption algorithm based on new 3-d chaos map. *Multimedia Tools and Applications*, 80, pp. 25583–25605.

Chen, J., Chen, L. and Zhou, Y., 2020. Cryptanalysis of a dna-based image encryption scheme. *Information Sciences*, 520, pp. 130–141.

Chen, J., Zhang, L. Y. and Zhou, Y., 2021. Re-evaluation of the security of a family of image diffusion mechanisms. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(12), pp. 4747–4758.

Chen, L., Ma, B., Zhao, X. and Wang, S., 2017. Differential cryptanalysis of a novel image encryption algorithm based on chaos and line map. *Nonlinear Dynamics*, 87(3), pp. 1797–1807.

Chen, L. and Wang, S., 2015. Differential cryptanalysis of a medical image cryptosystem with multiple rounds. *Computers in biology and medicine*, 65, pp. 69–75.

Daemen, J. and Rijmen, V., 2013. *The design of Rijndael: AES-the advanced encryption standard*. Springer Science & Business Media.

Das, S., Mondal, S. N. and Sanyal, M. K., 2018. Image encryption based on Arnold Cat map and GA operator. *Intelligent Engineering Informatics*. Springer. pp. 19–28.

Dawahdeh, Z. E., Yaakob, S. N. and bin Othman, R. R., 2018. A new image encryption technique combining elliptic curve cryptosystem with hill cipher. *Journal of King Saud University-Computer and Information Sciences*, 30(3), pp. 349–355.

Dhall, S., Pal, S. K. and Sharma, K., 2018. Cryptanalysis of image encryption scheme based on a new 1d chaotic system. *Signal processing*, 146, pp. 22–32.

Enayatifar, R., Abdullah, A. H. and Isnin, I. F., 2014. Chaos-based image encryption using a hybrid genetic algorithm and a DNA sequence. *Optics and Lasers in Engineering*, 56, pp. 83–93.

Enayatifar, R., Abdullah, A. H. and Lee, M., 2013. A weighted discrete imperialist competitive algorithm (WDICA) combined with chaotic map for image encryption. *Optics and Lasers in Engineering*, 51(9), pp. 1066–1077.

Essaid, M., Akharraz, I., Saaidi, A. et al., 2019. Image encryption scheme based on a new secure variant of hill cipher and 1d chaotic maps. *Journal of Information Security and Applications*, 47, pp. 173–187.

Fan, C. and Ding, Q., 2019. Analysing the dynamics of digital chaotic maps via a new period search algorithm. *Nonlinear Dynamics*, 97(1), pp. 831–841.

Fang, Q., Liu, Y. and Zhao, X., 2008. A chaos-based secure cluster protocol for wireless sensor networks. *Kybernetika*, 44(4), pp. 522–533.

FIPS, P., 2001. 180-2: Secure hash standard (shs)—computer security standard. *National Institute of Standards and Technology (NIST)*, .

Fridrich, J., 1998. Symmetric ciphers based on two-dimensional chaotic maps. *International Journal of Bifurcation and chaos*, 8(06), pp. 1259–1284.

Fu, C., Meng, W.-h., Zhan, Y.-f., Zhu, Z.-l., Lau, F. C., Chi, K. T. and Ma, H.-f., 2013. An efficient and secure medical image protection scheme based on chaotic maps. *Computers in biology and medicine*, 43(8), pp. 1000–1010.

Guesmi, R., Farah, M. A. B., Kachouri, A. and Samet, M., 2016. Hash key-based image encryption using crossover operator and chaos. *Multimedia tools and applications*, 75(8), pp. 4753–4769.

Hassan, M. A. S. and Abuhaiba, I. S. I., 2011. Image encryption using differential evolution approach in frequency domain. *Signal & Image Processing: An International Journal (SIPIJ)*, 2(1), pp. 51–69.

Holland, J. H., 1975. Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. *USA: University of Michigan*, .

Holland, J. H., 1992. Genetic algorithms. *Scientific american*, 267(1), pp. 66–73.

Hu, X., Wei, L., Chen, W., Chen, Q. and Guo, Y., 2020. Color image encryption algorithm based on dynamic chaos and matrix convolution. *IEEE access*, 8, pp. 12452–12466.

Hua, Z., Yi, S. and Zhou, Y., 2018. Medical image encryption using high-speed scrambling and pixel adaptive diffusion. *Signal Processing*, 144, pp. 134–144.

Hua, Z., Zhang, Y. and Zhou, Y., 2020. Two-dimensional modular chaotification system for improving chaos complexity. *IEEE Transactions on signal processing*, 68, pp. 1937–1949.

Hua, Z., Zhou, Y. and Bao, B., 2019. Two-dimensional sine chaotification system with hardware implementation. *IEEE Transactions on Industrial Informatics*, 16(2), pp. 887–897.

Huang, H., 2019. Novel scheme for image encryption combining 2d logistic-sine-cosine map and double random-phase encoding. *IEEE Access*, 7, pp. 177988–177996.

Katz, J. and Lindell, Y., 2020. *Introduction to modern cryptography*. CRC press.

Khan, M., 2015. A novel image encryption scheme based on multiple chaotic s-boxes. *Nonlinear Dynamics*, 82(1), pp. 527–533.

Kocarev, L. and Lian, S., 2011. *Chaos-based cryptography: Theory, algorithms and applications*. Vol. 354. Springer Science & Business Media.

Kumar, J. and Nirmala, S., 2012. Encryption of images based on genetic algorithm–a new approach. *Advances in Computer Science, Engineering & Applications*. Springer. pp. 783–791.

Lai, X. and Massey, J. L., 1990. A proposal for a new block encryption standard. *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, pp. 389–404.

Lee, W.-K., Phan, R. C.-W., Yap, W.-S. and Goi, B.-M., 2018. Spring: a novel parallel chaos-based image encryption scheme. *Nonlinear Dynamics*, 92(2), pp. 575–593.

Lestari, A. A., Suryadi, M., Ramli, K. et al., 2018. Modified logistic maps for discrete time chaos based random number generator. *2018 International Conference on Electrical Engineering and Computer Science (ICECOS)*. IEEE, pp. 391–396.

Li, C., Arroyo, D. and Lo, K.-T., 2010. Breaking a chaotic cryptographic scheme based on composition maps. *International Journal of Bifurcation and Chaos*, 20(08), pp. 2561–2568.

Li, C., Feng, B., Li, S., Kurths, J. and Chen, G., 2019. Dynamic analysis of digital chaotic maps via state-mapping networks. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 66(6), pp. 2322–2335.

Li, C., Li, S., Chen, G. and Halang, W. A., 2009. Cryptanalysis of an image encryption scheme based on a compound chaotic sequence. *Image and Vision Computing*, 27(8), pp. 1035–1039.

Li, C. and Lo, K.-T., 2011. Optimal quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks. *Signal processing*, 91(4), pp. 949–954.

Li, C., Tan, K., Feng, B. and Lü, J., 2021. The graph structure of the generalized discrete arnold's cat map. *IEEE Transactions on Computers*, 71(2), pp. 364–377.

Li, M., Wang, P., Liu, Y. and Fan, H., 2019. Cryptanalysis of a novel bit-level color image encryption using improved 1d chaotic map. *IEEE Access*, 7, pp. 145798–145806.

Li, S., Li, C., Chen, G., Bourbakis, N. G. and Lo, K.-T., 2008. A general quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks. *Signal Processing: Image Communication*, 23(3), pp. 212–223.

Li, Z., Peng, C., Li, L. and Zhu, X., 2018. A novel plaintext-related image encryption scheme using hyper-chaotic system. *Nonlinear Dynamics*, 94(2), pp. 1319–1333.

Liu, B., Xiang, H. and Liu, L., 2020. Reducing the dynamical degradation of digital chaotic maps with time-delay linear feedback and parameter perturbation. *Mathematical Problems in Engineering*, 2020(1), pp. 4926937.

Liu, L., Zhang, Z. and Chen, R., 2019. Cryptanalysis and improvement in a plaintext-related image encryption scheme based on hyper chaos. *IEEE Access*, 7, pp. 126450–126463.

Liu, Q. and Liu, L., 2020. Color image encryption algorithm based on dna coding and double chaos system. *IEEE Access*, 8, pp. 83596–83610.

Lone, P. N., Singh, D., Stoffová, V., Mishra, D. C., Mir, U. H. and Kumar, N., 2022. Cryptanalysis and improved image encryption scheme using elliptic curve and affine hill cipher. *Mathematics*, 10(20), pp. 3878.

Luo, Y., Liu, Y., Liu, J., Tang, S., Harkin, J. and Cao, Y., 2021. Counteracting dynamical degradation of a class of digital chaotic systems via unscented kalman filter and perturbation. *Information Sciences*, 556, pp. 49–66.

Masood, F., Ahmad, J., Shah, S. A., Jamal, S. S. and Hussain, I., 2020. A novel hybrid secure image encryption based on julia set of fractals and 3d lorenz chaotic map. *Entropy*, 22(3), pp. 274.

Matthews, R., 1989. On the derivation of a "chaotic" encryption algorithm. *Cryptologia*, 13(1), pp. 29–42.

May, R. M., 1974. Biological populations with nonoverlapping generations: stable points, stable cycles, and chaos. *Science*, 186(4164), pp. 645–647.

May, R. M., 1976. Simple mathematical models with very complicated dynamics. *Nature*, 261(5560), pp. 459–467.

Mozaffari, S., 2018. Parallel image encryption with bitplane decomposition and genetic algorithm. *Multimedia Tools and Applications*, 77, pp. 25799–25819.

Munir, N., Khan, M., Jamal, S. S., Hazzazi, M. M. and Hussain, I., 2021. Cryptanalysis of hybrid secure image encryption based on julia set fractals and three-dimensional lorenz chaotic map. *Mathematics and Computers in Simulation*, 190, pp. 826–836.

National Bureau of Standards, 1977. *Data Encryption Standard*. U.S. Department of Commerce, FIPS-Pub.46.

Nematzadeh, H., Enayatifar, R., Motameni, H., Guimarães, F. G. and Coelho, V. N., 2018. Medical image encryption using a hybrid model of modified genetic algorithm and coupled map lattices. *Optics and Lasers in Engineering*, 110, pp. 24–32.

Pak, C., An, K., Jang, P., Kim, J. and Kim, S., 2019. A novel bit-level color image encryption using improved 1d chaotic map. *Multimedia Tools and Applications*, 78(9), pp. 12027–12042.

Pak, C., Kim, J., Pang, R., Song, O., Kim, H., Yun, I. and Kim, J., 2021. A new color image encryption using 2d improved logistic coupling map. *Multimedia Tools and Applications*, 80, pp. 25367–25387.

Patro, K. A. K., Soni, A., Netam, P. K. and Acharya, B., 2020. Multiple grayscale image encryption using cross-coupled chaotic maps. *Journal of Information Security and Applications*, 52, pp. 102470.

Petitcolas, F. A., 2023. Kerckhoffs' principle. *Encyclopedia of Cryptography, Security and Privacy*. Springer. pp. 1–2.

Ping, P., Xu, F., Mao, Y. and Wang, Z., 2018. Designing permutation–substitution image encryption networks with henon map. *Neurocomputing*, 283, pp. 53–63.

Premkumar, R. and Anand, S., 2018. Secured and compound 3-D chaos image encryption using hybrid mutation and crossover operator. *Multimedia Tools and Applications*, pp. 1–17.

Rajaraman, V., 2016. Ieee standard for floating point numbers. *Resonance*, 21(1), pp. 11–30.

Ravichandran, D., Praveenkumar, P., Rayappan, J. B. B. and Amirtharajan, R., 2016. Chaos based crossover and mutation for securing DICOM image. *Computers in biology and medicine*, 72, pp. 170–184.

Rehman, A. U., Wang, H., Shahid, M. M. A., Iqbal, S., Abbas, Z. and Firdous, A., 2019. A selective cross-substitution technique for encrypting color images using chaos, dna rules and sha-512. *IEEE Access*, 7, pp. 162786–162802.

Shannon, C. E., 1949. Communication theory of secrecy systems. *The Bell system technical journal*, 28(4), pp. 656–715.

Singh, L. D., Thingbaijam, R., Patgiri, R. and Singh, K. M., 2024. Cryptanalysis of cross-coupled chaotic maps multi-image encryption scheme. *Journal of Information Security and Applications*, 80, pp. 103694.

Sivanandam, S. and Deepa, S., 2008. Genetic algorithms. *Introduction to genetic algorithms*. Springer. pp. 15–37.

Solak, E., Cokal, C., Yildiz, O. T. and Biyikoğlu, T., 2010. Cryptanalysis of fridrich's chaotic image encryption. *International Journal of Bifurcation and Chaos*, 20(05), pp. 1405–1413.

Teng, L., Wang, X. and Xian, Y., 2022. Image encryption algorithm based on a 2d-clss hyperchaotic map using simultaneous permutation and diffusion. *Information Sciences*, 605, pp. 71–85.

Tong, X. and Cui, M., 2008. Image encryption with compound chaotic sequence cipher shifting dynamically. *Image and Vision Computing*, 26(6), pp. 843–850.

Wang, B., Xie, Y., Zhou, C., Zhou, S. and Zheng, X., 2016. Evaluating the permutation and diffusion operations used in image encryption based on chaotic maps. *Optik*, 127(7), pp. 3541–3545.

Wang, X. and Guo, K., 2014. A new image alternate encryption algorithm based on chaotic map. *Nonlinear dynamics*, 76(4), pp. 1943–1950.

Wang, X. and Xu, D., 2014. Image encryption using genetic operators and intertwining logistic map. *Nonlinear Dynamics*, 78(4), pp. 2975–2984.

Wang, Y., Zhao, Y., Zhou, Q. and Lin, Z., 2018. Image encryption using partitioned cellular automata. *Neurocomputing*, 275, pp. 1318–1332.

Wen, H., Chen, R., Yang, J., Zheng, T., Wu, J., Lin, W., Jian, H., Lin, Y., Ma, L., Liu, Z. et al., 2024. Security analysis of a color image encryption based on bit-level and chaotic map. *Multimedia Tools and Applications*, 83(2), pp. 4133–4149.

Wen, H., Lin, Y. and Feng, Z., 2024. Cryptanalyzing a bit-level image encryption algorithm based on chaotic maps. *Engineering Science and Technology, an International Journal*, 51, pp. 101634.

Wen, H., Lin, Y., Yang, L. and Chen, R., 2024. Cryptanalysis of an image encryption scheme using variant hill cipher and chaos. *Expert Systems with Applications*, 250, pp. 123748.

Wen, W., Zhang, Y., Su, M., Zhang, R., Chen, J.-x. and Li, M., 2017. Differential attack on a hyper-chaos-based image cryptosystem with a classic bi-modular architecture. *Nonlinear Dynamics*, 87(1), pp. 383–390.

Wong, K.-W., Yap, W.-S., Goi, B.-M. and Wong, D. C.-K., 2020. A new chaotic map based on logistic and beta maps. *Cryptology and Information Security Conference 2020*, p. 65.

Wong, K.-W., Yap, W.-S., Wong, D. C.-K., Phan, R. C.-W. and Goi, B.-M., 2020. Cryptanalysis of genetic algorithm-based encryption scheme. *Multimedia Tools and Applications*, 79(35), pp. 25259–25276.

Wu, J., Liao, X. and Yang, B., 2018a. Cryptanalysis and enhancements of image encryption based on three-dimensional bit matrix permutation. *Signal Processing*, 142, pp. 292–300.

Wu, J., Liao, X. and Yang, B., 2018b. Image encryption using 2d hénon-sine map and dna approach. *Signal processing*, 153, pp. 11–23.

Wu, Q., Zhang, F., Hong, Q., Wang, X. and Zeng, Z., 2021. Research on cascading high-dimensional isomorphic chaotic maps. *Cognitive Neurodynamics*, 15, pp. 157–167.

Wu, Y., Noonan, J. P., Agaian, S. et al., 2011. Npcr and uaci randomness tests for image encryption. *Cyber journals: multidisciplinary journals in science and technology, Journal of Selected Areas in Telecommunications (JSAT)*, 1(2), pp. 31–38.

Xie, E. Y., Li, C., Yu, S. and Lü, J., 2017. On the cryptanalysis of fridrich's chaotic image encryption scheme. *Signal processing*, 132, pp. 150–154.

Xu, L., Li, Z., Li, J. and Hua, W., 2016. A novel bit-level image encryption algorithm based on chaotic maps. *Optics and Lasers in Engineering*, 78, pp. 17–25.

Yap, W.-S. and Phan, R. C.-W., 2017. Commentary on "a block chaotic image encryption scheme based on self-adaptive modelling"[applied soft computing 22 (2014) 351–357]. *Applied Soft Computing*, 52, pp. 501–504.

Yap, W.-S., Phan, R. C.-W., Goi, B.-M., Yau, W.-C. and Heng, S.-H., 2016. On the effective subkey space of some image encryption algorithms using external key. *Journal of Visual Communication and Image Representation*, 40, pp. 51–57.

Yap, W.-S., Phan, R. C.-W., Yau, W.-C. and Heng, S.-H., 2015. Cryptanalysis of a new image alternate encryption algorithm based on chaotic map. *Nonlinear Dynamics*, 80(3), pp. 1483–1491.

Ye, G. and Zhou, J., 2014. A block chaotic image encryption scheme based on self-adaptive modelling. *Applied Soft Computing*, 22, pp. 351–357.

Zahmoul, R., Ejbali, R. and Zaied, M., 2017. Image encryption based on new beta chaotic maps. *Optics and Lasers in Engineering*, 96, pp. 39–49.

Zhang, G., Ding, W. and Li, L., 2020. Image encryption algorithm based on tent delay-sine cascade with logistic map. *Symmetry*, 12(3), pp. 355.

Zhang, G., Zhang, F., Liao, X., Lin, D. and Zhou, P., 2017. On the dynamics of new 4d lorenz-type chaos systems. *Advances in Difference Equations*, 2017(1), pp. 1–13.

Zhang, L.-b., Zhu, Z.-l., Yang, B.-q., Liu, W.-y., Zhu, H.-f. and Zou, M.-y., 2015. Cryptanalysis and improvement of an efficient and secure medical image protection scheme. *Mathematical Problems in Engineering*, 2015(1), pp. 913476.

Zhang, L. Y., Li, C., Wong, K.-W., Shu, S. and Chen, G., 2012. Cryptanalyzing a chaos-based image encryption algorithm using alternate structure. *Journal of Systems and Software*, 85(9), pp. 2077–2085.

Zhang, L. Y., Liu, Y., Pareschi, F., Zhang, Y., Wong, K.-W., Rovatti, R. and Setti, G., 2018. On the security of a class of diffusion mechanisms for image encryption. *IEEE Transactions on Cybernetics*, 48(4), pp. 1163–1175.

Zhang, W., Yu, H., Zhao, Y.-l. and Zhu, Z.-l., 2016. Image encryption based on three-dimensional bit matrix permutation. *Signal Processing*, 118, pp. 36–50.

Zhang, Y.-Q., He, Y., Li, P. and Wang, X.-Y., 2020. A new color image encryption scheme based on 2dnlcml system and genetic operations. *Optics and Lasers in Engineering*, 128, pp. 106040.

Zhang, Y., Wang, Y. and Shen, X., 2007. A chaos-based image encryption algorithm using alternate structure. *Science in China Series F: Information Sciences*, 50(3), pp. 334–341.

Zhou, G., Zhang, D., Liu, Y., Yuan, Y. and Liu, Q., 2015. A novel image encryption algorithm based on chaos and Line map. *Neurocomputing*, 169, pp. 150–157.

Zhou, R. and Yu, S., 2024. Break an enhanced plaintext-related chaotic image encryption algorithm. *Chaos, Solitons & Fractals*, 181, pp. 114623.

Zhou, Y., Bao, L. and Chen, C. P., 2014. A new 1d chaotic system for image encryption. *Signal processing*, 97, pp. 172–182.

Zhou, Y., Cao, W. and Chen, C. P., 2014. Image encryption using binary bitplane. *Signal processing*, 100, pp. 197–207.

Zhu, H., Zhao, Y. and Song, Y., 2019. 2d logistic-modulated-sine-coupling-logistic chaotic map for image encryption. *IEEE Access*, 7, pp. 14081–14098.

LIST OF PUBLICATIONS

Journal:

1. Wong, K.W., Yap, W.S., Wong, D.C.K., Phan, R.C.W. and Goi, B.M., 2020. Cryptanalysis of genetic algorithm-based encryption scheme. Multimedia Tools and Applications, 79, pp.25259-25276. (Scopus Q1, Web of Science Q2)

2. Wong, K.W., Yap, W.S., Goi, B.M., Wong, D.C.K. and Ye, G., 2024. Cryptanalysis of an image encryption scheme based on two-point diffusion strategy and Henon map. Journal of Information Security and Applications, 81, p.103692. (Scopus Q1, Web of Science Q2)

3. Cross-plane color image encryption based on two-dimensional sine-henon map and genetic algorithm. (Submitted to Signal Processing: Image Communication)

Conference:

1. Wong, K.W., Yap, W.S., Goi, B.M. and Wong, D.C., 2019, April. Differential cryptanalysis on chaotic based image encryption scheme. In IOP conference series: materials science and engineering (Vol. 495, No. 1, p. 012041). IOP Publishing.

2. Wong, K.W., Yap, W.S., Goi, B.M. and Wong, D.C.K., A New Chaotic Map Based on Logistic and Beta Maps. In Cryptology and Information Security Conference 2020 (p. 65).

3. Wong, K.W., Yap, W.S., Goi, B.M. and Wong, D.C.K., 2022, February. A New Image Encryption Scheme Based on Hyperchaotic System and SHA-2. In Proceedings of the 6th International Conference on Digital Signal Processing (pp. 140-145).

4. Wong, K.W., Yap, W.S., Goi, B.M. and Wong, D.C., 2023, February. Pseudorandom Number Generator Based on Two-Dimensional Improved Modular Chaotic Map. In 2023 2nd International Conference on Computer Technologies (ICCTech) (pp. 12-16). IEEE.