# DEVELOPMENT OF EFFICIENT COLOR IMAGE ENCRYPTION TECHNIQUES USING EVOLUTIONARY APPROACHES

A thesis submitted

in partial fulfillment of the requirements for the award of degree of

**DOCTOR OF PHILOSOPHY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING DEPARTMENT**


Submitted By

**MANJIT KAUR**

**(901503012)**

Under the supervision of

**DR. VIJAY KUMAR**

(Assistant Professor)




THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY

PATIALA-147004, PUNJAB, INDIA.


**AUGUST, 2019**

# Certificate

I, Manjit Kaur, Regn. No. 901503012, hereby declares that the thesis entitled **"Development of Efficient Color Image Encryption Techniques Using Evolutionary Approaches"** submitted to the Computer Science and Engineering Department at Thapar Institute of Engineering and Technology, Patiala, Punjab, India is an authenticated record of my own work for the award of the degree of "Doctor of Philosophy" under the supervision of Dr. Vijay Kumar. This report has not been submitted to any other institution for the award of any other degree.

Manjit Kaur
Regn. No. 901503012

Place: Patiala
Date: 13.08.2019

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Verified by:
13/8/19
Dr. Vijay Kumar
Assistant Professor
Computer Science and Engineering Department
Thapar Institute of Engineering and Technology,
Patiala, Punjab, India.

# Abstract

The advancements in multimedia applications are rapidly increasing nowadays. A number of confidential images are transferred over the public networks day-by-day. Therefore, secure transmission of images has become a significant research area. Several image encryption techniques have been designed to achieve the particular necessities raised by different users. The image encryption techniques convert confidential image into a noisy image using secret key. The actual image is recovered if and only if the receiver has an authenticated secret key. Various techniques such as chaotic maps, evolutionary, DNA, cellular automata, *etc.* have been used to encrypt the images.

Image encryption differs from text encryption as a result of few numbers of inherent characteristics such as huge size and redundancy. The traditional text encryption techniques fail to handle huge size and redundancy of images. The main challenges of image encryption are robustness against attacks, keyspace, key sensitivity, and diffusion. The chaotic maps are extensively utilized in the field of image encryption to generate the secret keys. However, these maps suffer from parameter tuning problem. Recent studies have shown that the improper selection of parameter values makes secret keys generated from chaotic system vulnerable. Therefore, many meta-heuristic techniques have been introduced in the literature for image encryption to improve the selection of chaotic system's parameters. However, these techniques suffer from poor computational speed. Also, designing an efficient multi-objective fitness function is still a challenging issue. To overcome these issues, meta-heuristic based image encryption techniques are proposed in this thesis.

An efficient image encryption technique in nonsubsampled contourlet transform using genetic algorithm (IGN) is proposed. Initially, nonsubsampled contourlet transform is utilized to decompose the input image into sub-bands. The beta chaotic map is used to develop a pseudo-random key that encrypts the coefficients of sub-bands. A multi-objective fitness function is designed to find the optimal parameters for beta chaotic map. Inverse nonsubsampled contourlet transform is performed to obtain an encrypted image. The performance of IGN is compared with well-known meta-heuristic based image encryption techniques. Experimental results reveal that IGN provides better computational speed and high encryption intensity.

However, genetic algorithm may suffers from local optima and pre-mature convergence issues. To deal with these issues, an image encryption technique using differential evolu-

tion in nonsubsampled contourlet transform (so called IDN) is proposed. In this technique, two new concepts are utilized to encrypt the images in an efficient manner. The first one is Arnold transform, which is used to permute the pixel position of an input image to generate a scrambled image. The second one is differential evolution, which is used to tune the parameters required by a beta chaotic map. The entropy of an encrypted image is used as a fitness function. IDN is compared with seven well-known image encryption techniques over five well-known benchmark test images. The experimental results reveal that IDN outperforms the existing techniques in terms of security and better visual quality.

To improve the local search ability of IDN, the initial conditions of an intertwining logistic map are generated by utilizing memetic differential evolution and Arnold transform (IIMA). It utilizes local chaotic search to improve the search ability of standard differential evolution. Initially, the color image is decomposed into red, green, and blue channels. Arnold transform is used to shuffle the pixel position of all three channels to develop the shuffled channels. Afterward, memetic differential evolution is used to optimize the parameters required by intertwining logistic map. The correlation coefficient and entropy are used as a fitness function. The intertwining map generates the secret keys to encrypt the shuffled color channels. The encrypted color channels are combined to obtain the encrypted image. Extensive experiments are carried out by considering IIMA and the existing competitive image encryption techniques. Experimental results reveal that IIMA provides higher efficiency and security as compared to the existing image encryption techniques.

To secure the secret key, an efficient Image encryption technique based on Secure hash algorithm (SHA-3), adaptive differential evolution, and Lorenz-like chaotic system known as ISAL is designed. ISAL utilizes SHA-3 along with adaptive differential evolution to reduce the issues associated with Lorenz-like chaotic system. In ISAL, adaptive differential evolution is used to optimize the input parameters for Lorenz-like chaotic system. SHA-3 is used to generate a secret key based on the input image. The optimized parameters and external secret keys are used to generate the initial values for Lorenz-like chaotic system that make it sensitive to an input image and provide resistance against both known-plaintext and known-ciphertext attacks. ISAL is compared with five well-known image encryption techniques over four color benchmark test images. The experimental results reveal that ISAL outperforms the existing techniques in terms of security and quality measures. The noise and enhancement attacks are also applied to test the robustness of ISAL.

Although, IGN, IDN, IIMA, and ISAL provide better encryption results than the existing techniques. But, they suffer from poor computational speed especially in case of high-resolution images. Therefore, an Image encryption technique based on Fourier-Mellin moments and intertwining logistic map (IFIM) is proposed. It uses multi-objective non-dominated sorting genetic algorithm based on reinforcement learning (MNSGA-RL) to optimize the required parameters of intertwining logistic map. Fourier-Mellin moments are used to make the secret keys more secure. Thereafter, permutation and diffusion operations are carried out on the input image using secret keys. The performance of IFIM is eval-

uated on five well-known benchmark images and also compared with seven well-known existing encryption techniques. The experimental results reveal that IFIM outperforms the others in terms of entropy, correlation analysis, unified average changing intensity, and number of pixel change rate. The simulation results reveal that IFIM provides a high level of security and robustness against various types of attacks. To improve the computational speed, IFIM is implemented in a parallel fashion using master-slave environment. The run time analysis has been done to determine the computationally expensive operations. Thereafter, IFIM operations are divided into master and slave jobs. Message passing interface (MPI) is used for intercommunication between master and slave nodes. The simulation results show that parallel IFIM provides a significant improvement in computational speed as compared to the existing techniques and sequential IFIM.

The proposed techniques (*i.e.,* IGN, IDN, IIMA, ISAL, and IFIM) satisfy the various statistical parameters and offer tangible resistance to differential, occlusion, and chosen plaintext attacks on gray as well as color images. The proposed techniques achieve not only a desirable level of security, but, also high efficiency and better computational speed. Therefore, the designed techniques are well suitable for real-life applications.

# Acknowledgments

Firstly, my sincere thanks to Guru Granth Sahib ji for giving me the strength and unlimited mercies to complete this research work.

I would like to express my sincere gratitude to my supervisor Dr. Vijay Kumar for his patience, motivation, research expertise, and support. He always gave the freedom to pursue varied ideas and helped me enhance my research skills. His guidance helped me to improve my research methodology and writing skills.

Besides my supervisor, I would like to thank other faculty members, Prof. Maninder Singh and Dr. Parashant Singh Rana of the Computer Science and Engineering Department who provided with valuable feedback during the various interactions and presentations in Thapar Institute of Engineering and Technology, Patiala, India. My sincere thanks to Prof. Mukesh Singh and Dr. Sharad Saxena for providing useful feedback and suggestions on the image encryption techniques used in this research. I am thankful to the entire non-teaching faculty of Computer Science and Engineering Department for providing high-end computational infrastructure for the experiments.

I am very thankful to my parents for their encouragement and sacrifices, and I wish to mention a very special acknowledgment to Dr. Dilbag Singh, my husband, for his continued support and co-operation. I pay regard to one and everyone who knowingly or unknowingly supported me during this journey of knowledge.

Finally, I want to bestow all my love to my world, my son, Anveen Singh Gill, who came into my life during this journey and showed me in the path of positiveness with his heavenly and divine smile. Love you son.

Manjit Kaur

(901503012)

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Preamble

With the advancement in computer applications and internet technology, security of digital contents becomes a challenging issue [1]. In recent days, digital images have turned out to be a significant way to communicate and save the potential information [2]. Thus, digital image has penetrated into all fields of our life. Images play an important role in numerous applications such as medical imaging, military, telemedicine, remote sensing, *etc.* [3]. Secure encryption techniques are required to prevent the unauthorized stealing and modify the transmitted image [4]. Images are different from the text due to the intrinsic characteristics such as correlation between adjacent pixels, huge space, higher redundancy, *etc.* Therefore, the existing well-known text encryption techniques become computationally expensive and prone to security attacks in the case of images [5].

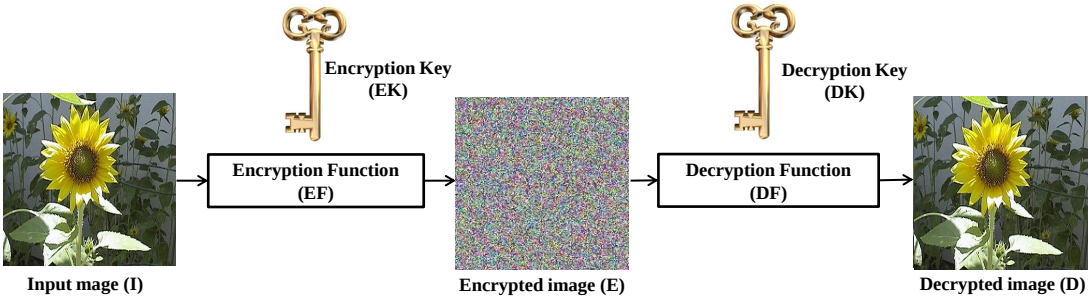



Figure 1.1: Generic framework of image encryption

Figure 1.1 illustrates the general framework of an image encryption. An input image that needs to be encrypted is called a plain-image ($I$). The encryption process is demon-

strated as:

$$E = EF_{EK}(I) \tag{1.1}$$

where $EF()$ is the encryption function applied on $I$ using encryption key $(EK)$. $E$ represents an encrypted image. The encrypted image $(E)$ looks like a noisy image, which hides the original information. Similarly, at the receiver end, the encrypted image is decrypted using the decryption key $(DK)$ and decryption function $(DF())$ as:

$$D = DF_{DK}(E) \tag{1.2}$$

Here, $D$ is the decrypted image. It is observed from Figure 1.1 that $I$ and $D$ are identical to each other.

The image encryption techniques are broadly classified into two main categories such as symmetric and asymmetric. In case of symmetric image encryption, the encryption and decryption keys are same, *i.e.,* $EK = DK$. The keys are to be kept secret during the communication. When different keys are used for encryption and decryption, the image encryption is known as asymmetric image encryption, *i.e.,* $EK \neq DK$. In this system, $EK$ is taken as public and $DK$ is kept private.

## 1.2 Chaotic maps

Chaotic maps are well studied in a dynamic environment as they exhibit chaotic behavior [6]. It means that a small change in initial conditions can produce drastic changes in their outputs. These maps are categorized into two categorizes such as discrete and continuous maps [7]. They are extensively utilized in secure communication and considered a favourable trade-off between security and computational speed. Chaos based encryption technique possesses several features such as sensitivity to initial circumstances, determinacy, and ergodicity [8].

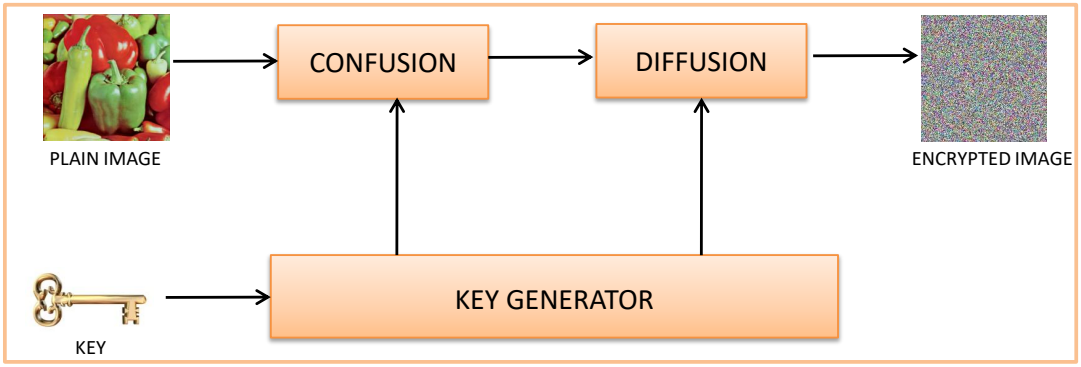



Figure 1.2: Image encryption using chaotic map

Figure 1.2 depicts the process of chaos based image encryption technique. The chaotic maps divide an encryption technique into dual phases, namely diffusion and permutation [9]. The permutation or confusion phase shuffles the image pixels using pseudo-random sequence. In diffusion phase, the pixel values of permuted image are modified [10]. In real time, cryptosystem developers may integrate permutation and diffusion to attain a significant computational security [11].

Table 1.1 shows the variants of chaotic map that have been used in the field of image encryption. The categorization of these variants has been done by considering the time domain, spatial domain, number of dimensions (Dim), and number of parameters (Param). It is observed that each map requires a different number of parameters to develop a secure key.

Table 1.1: Variants of chaotic map

| Ref. | Chaotic maps | Time domain | Space domain | Dim | Param |
|---|---|---|---|---|---|
| [12] | 2-D Rational map | Discrete | Rational | 2 | 2 |
| [13] | 3-cells CNN system | Continuous | Real | 3 | 4 |
| [14] | Arnold's cat map | Discrete | Real | 2 | 0 |
| [15, 16] | Baker's map | Discrete | Real | 2 | 0 |
| [17, 18] | Chen attractor | Continuous | Real | 3 | 2 |
| [19, 20] | Circle map | Discrete | Real | 1 | 2 |
| [21] | Exponential map | Discrete | Complex | 2 | 1 |
| [22, 23] | Hyper Logistic map | Discrete | Real | 2 | 3 |
| [24] | Hénon map | Discrete | Real | 2 | 2 |
| [25] | Logistic map | Discrete | Real | 1 | 1 |
| [26] | Lorenz attractor | Continuous | Real | 3 | 3 |
| [27] | Tent map | Discrete | Real | 1 | 1 |
| [28, 29] | Tinkerbell map | Discrete | Real | 2 | 4 |
| [30, 31] | Zaslavskii map | Discrete | Real | 2 | 4 |

The subsequent sections present the detail description of different chaotic maps which are used in this research work to develop secret keys for image encryption.

### 1.2.1 Beta chaotic map

The beta chaotic map is newly introduced into the family of chaotic maps by Samuel *et al.* [32]. The mathematical representation of beta chaotic map is defined as:

$$y_{n+1} = t \times Beta(y_n; y_1, y_2, u, v) \tag{1.3}$$

where

$$Beta(y;u,v,y_1,y_2) = \begin{cases} \left(\frac{(y-y_1)}{(y_c-y_1)}\right)^u \left(\frac{(y_2-y)}{(y_2-y_c)}\right)^v & \text{if} \quad y \in (y_1,y_2) \\ 0 & \text{otherwise.} \end{cases} \tag{1.4}$$

Given $u$, $v$, $y_1$ and $y_2 \in R_n$, $y_1 < y_2$, one has

$$y_c = \frac{(uy_2 + vy_1)}{u+v} \tag{1.5}$$

$$u = l_1 + m_1 \times b \tag{1.6}$$

and

$$v = l_2 + m_2 \times b \tag{1.7}$$

where $b$ represents the bifurcation parameter. $t$ controls the amplitude of chaotic map. $l_1$, $m_1$, $l_2$, and $m_2$ are constants.

Beta chaotic map is sensitive toward the initial conditions and variation in bifurcation parameter. Hence, it enhances the security of encryption procedures against various attacks [33].

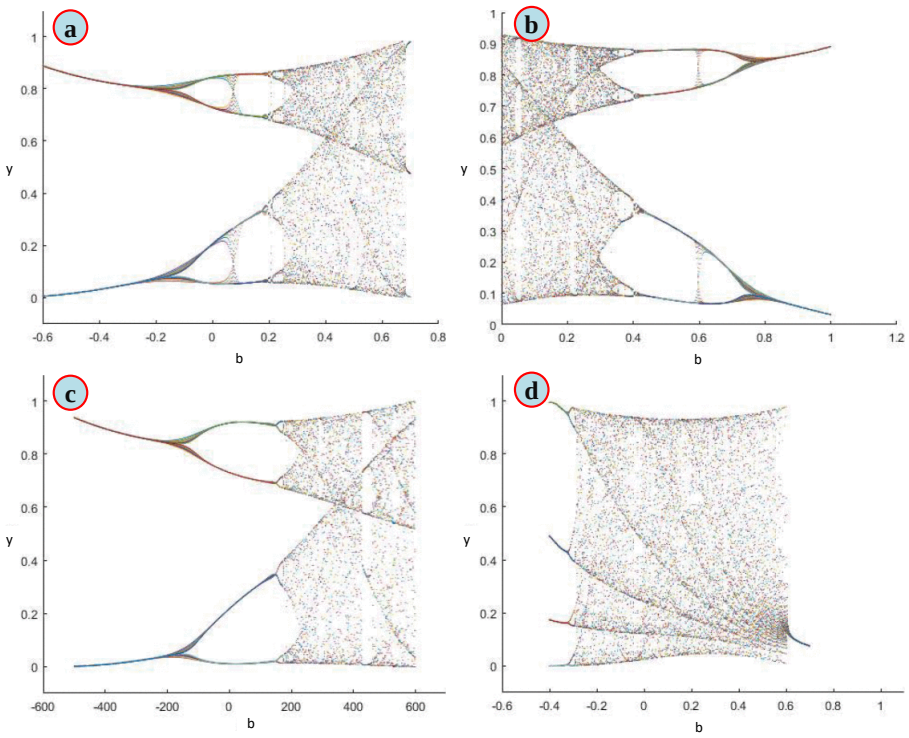



Figure 1.3: Range of bifurcation parameters of beta chaotic map

Figure 1.3 shows different beta maps with their respective bifurcation range. In Figure 1.3(a), the following parameters are used. $b = [-0.8 : 0.7]$, $y_1 = -1$, $y_2 = 1$, $t = 0.85$, $l_1 = 5$, $m_1 = 1$, $l_2 = 3$, and $m_2 = -1$. It has been observed that beta chaotic map behaves

like a logistic map. In Figure 1.3(b), the parameters are set as follows $b = [0 : 1]$, $y_1 = -1$, $y_2 = 1$, $t = 0.88$, $l_1 = 5$, $m_1 = -1.5$, $l_2 = 2$, and $m_2 = 0.5$. It shows the inverted behavior of logistic map. In Figure 1.3(c), the following parameters are set as follows $b = [500 : 600]$, $y_1 = -1$, $y_2 = 1$, $t = 0.9$, $l_1 = 5$, $m_1 = 0.01$, $l_2 = 3$, and $m_2 = -0.01$. It is observed that the bifurcation parameter has a large range. In Figure 1.3(d), the value of parameters are set as follows. $b = [-0.4 : 0.7]$, $y_1 = -0.7$, $y_2 = 1$, $t = 0.93$, $l_1 = 8$, $m_1 = 1$, $l_2 = 3$, and $m_2 = -1$. The behavior of beta map looks alike of standard chaotic map. Therefore, beta chaotic map has a significant bifurcation range.

### 1.2.2 Intertwining logistic map

3-D intertwining logistic map was proposed by Sam and Devraj in 2012 [34]. It provides better chaotic behavior than the classical one-dimensional (1-D) logistic map. The classical logistic map contains only one control parameter and one initial condition. It generates weak secret keys that can be easily compromised [35, 36]. But, the intertwining logistic map contains more initial and control parameters as compared to the classical one . Moreover, it also alleviates the problems of stable windows, blank windows, and uneven distribution of iterated sequences that exists in classical logistic map. Hence, it can generate the strong secret keys. The intertwining logistic map is defined as [34, 37]:

$$\begin{aligned} x_{i+1} &= [\alpha \times a_1 \times y_i \times (1 - x_i) + z_i] \mod 1 \\ y_{i+1} &= [\alpha \times a_2 \times y_i + z_i \times \frac{1}{(1 + x_{i+1}^2)}] \mod 1 \\ z_{i+1} &= [\alpha \times (x_{i+1} + y_{i+1} + a_3) \times sin(z_i)] \mod 1 \end{aligned} \tag{1.8}$$

Here, $\alpha$ represents the control parameter. $a_1$, $a_2$, and $a_3$ represent constant parameters. $x_i$, $y_i$, and $z_i$ represent initial states. $x_{i+1}$, $y_{i+1}$, and $z_{i+1}$ represent the random sequences that generated through initial states. The value of parameters should satisfy the following constraints $|a_1| > 33.5$, $|a_2| > 37.9$, $|a_3| > 35.7$, and $0 < \alpha \leq 3.999$.

### 1.2.3 Lorenz-like chaotic system

The first non-linear dynamic system was proposed by Edward Lorenz [38]. It is known as Lorenz chaotic system because it exhibits properties like chaotic systems. It is computed as [39]:

$$\begin{aligned} \dot{x} &= a(y - x) \\ \dot{y} &= (b - z)x - y \\ \dot{z} &= xy - cz \end{aligned} \tag{1.9}$$

Here, $a$ and $c$ are real constant parameters of Lorenz-like chaotic system. $b$ shows bifurcation parameter. $x$, $y$, and $z$ are state parameters. The time derivatives of state variables are represented by $\dot{x}$, $\dot{y}$, and $\dot{z}$, respectively. Figure 1.4 shows the chaotic attractor of Lorenz chaotic system for $a = 10$, $b = 28$, and $c = 8/3$.

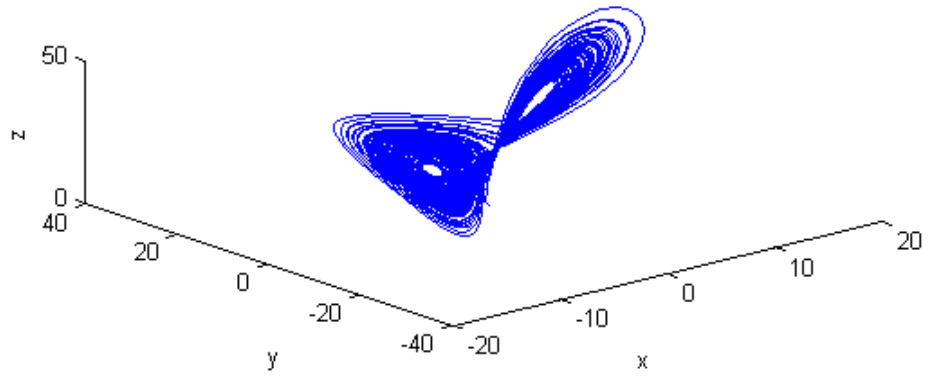



Figure 1.4: Chaotic attractor of Lorenz chaotic system

Huang *et al.* [40] modified Edward's system by removing $y$ variable from Eq. (1.9). Then, the modified Lorenz chaotic map can be written as [41, 42]:

$$\begin{aligned} \dot{x} &= a(y - x) \\ \dot{y} &= bx - zx \\ \dot{z} &= xy - cz \end{aligned} \tag{1.10}$$

When $a = 10$, $b = 50$, and $c = 8/3$, the chaotic attractor of system using Eq. (1.10) is shown in Figure 1.5.

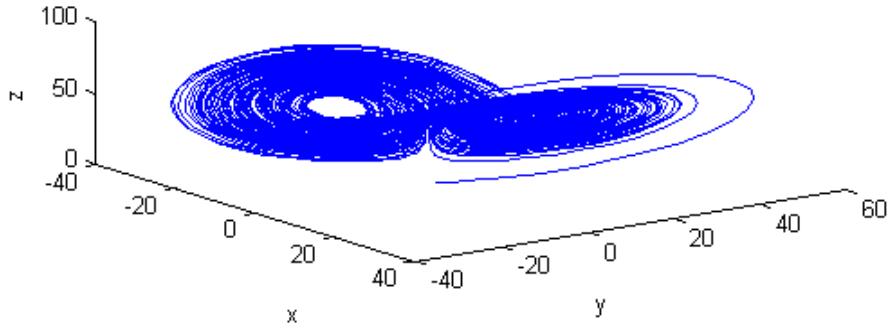



Figure 1.5: Chaotic attractor of modified Lorenz chaotic system

The bifurcation parameter $(b)$ is constant in both systems. Due to this, the equilibrium points are also fixed [43, 44]. Huang *et al.* [45] proposed a new Lorenz-like chaotic

system with varying bifurcation parameter to improve the complex behavior of chaotic system. Lorenz-like chaotic system can be defined as:

$$\begin{aligned}
\dot{x} &= a(y - x) \\
\dot{y} &= bx - zx \\
\dot{z} &= xy - cz \\
b &= \begin{cases} l_1 + l_2, & |x| > \delta \\ l_1 - l_2, & |x| < \delta \end{cases}
\end{aligned} \tag{1.11}$$

Here, $l_1$, $l_2$, and $\delta$ represent real constant parameters. $b$ switches between $l_1 + l_2$ and $l_1 - l_2$ depend on the value of $x$. Due to this variation, chaotic system will generate dense chaotic attractors which show more complex behavior than the above-mentioned chaotic systems (see Figure 1.6). $\delta$ influences the dynamic behavior of system significantly [43].

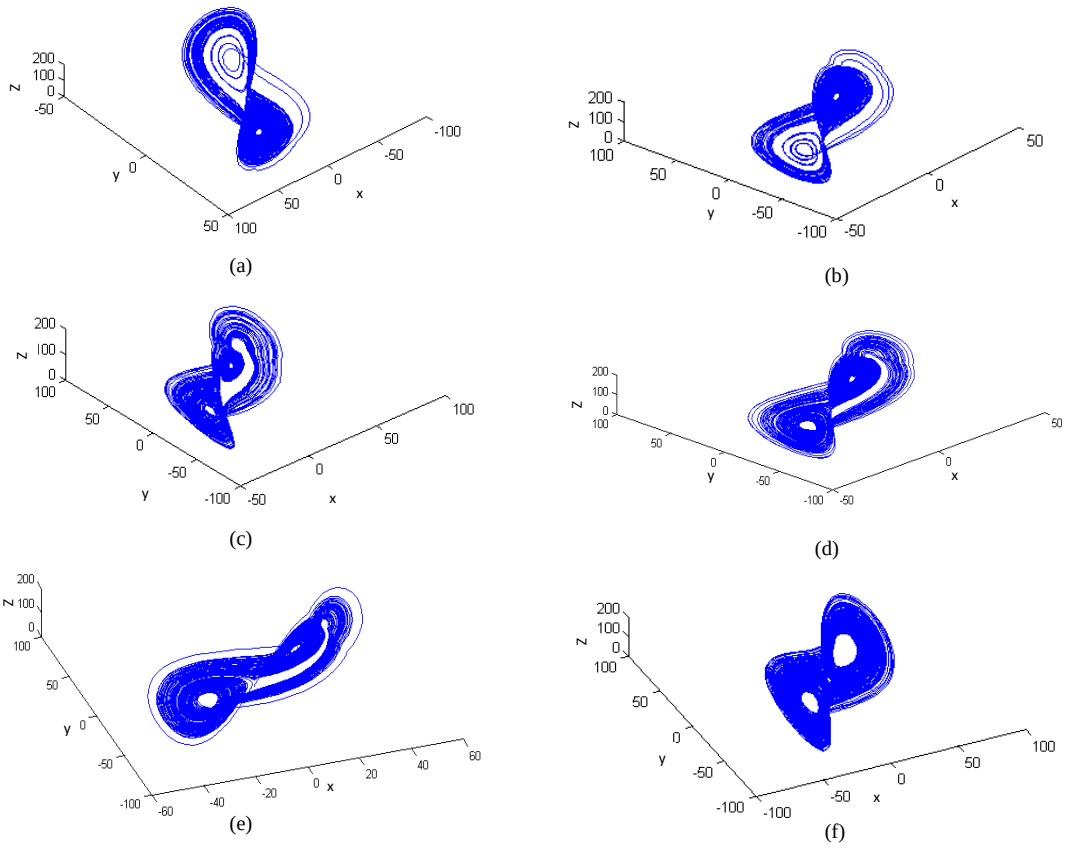



Figure 1.6: Chaotic attractors of Lorenz-like chaotic system when (a) $\delta = 4$, (b) $\delta = 10$, (c) $\delta = 20$, (d) $\delta = 30$, (e) $\delta = 40$, and (f) $\delta = 50$

Figure 1.6 shows different chaotic attractors of the system at different values of $\delta =$

$4, 10, 20, 30, 40$, and $50$. To show chaotic attractors, we choose $a = 20$, $c = 8$, $l_1 = 70$, and $l_2 = 15$. The state variables $x_0$, $y_0$, and $z_0$ are initialized with $1$.

## 1.3 Meta-heuristic techniques

Meta-heuristic techniques play a significant role to optimize NP-Hard problems. The advantages of these techniques are used to optimize the constant parameters required by encryption process [46]. The subsequent subsections present meta-heuristic techniques that are used in this research work to optimize the parameters of chaotic maps.

### 1.3.1 Genetic algorithm

Genetic algorithm (GA) is a well-known evolutionary technique to find optimal solutions for various Non-deterministic polynomial time (NP) complete problems. The image encryption technique can be considered as NP-complete problem. Therefore, GA is used to find the optimal parameter setting for efficient encryption.

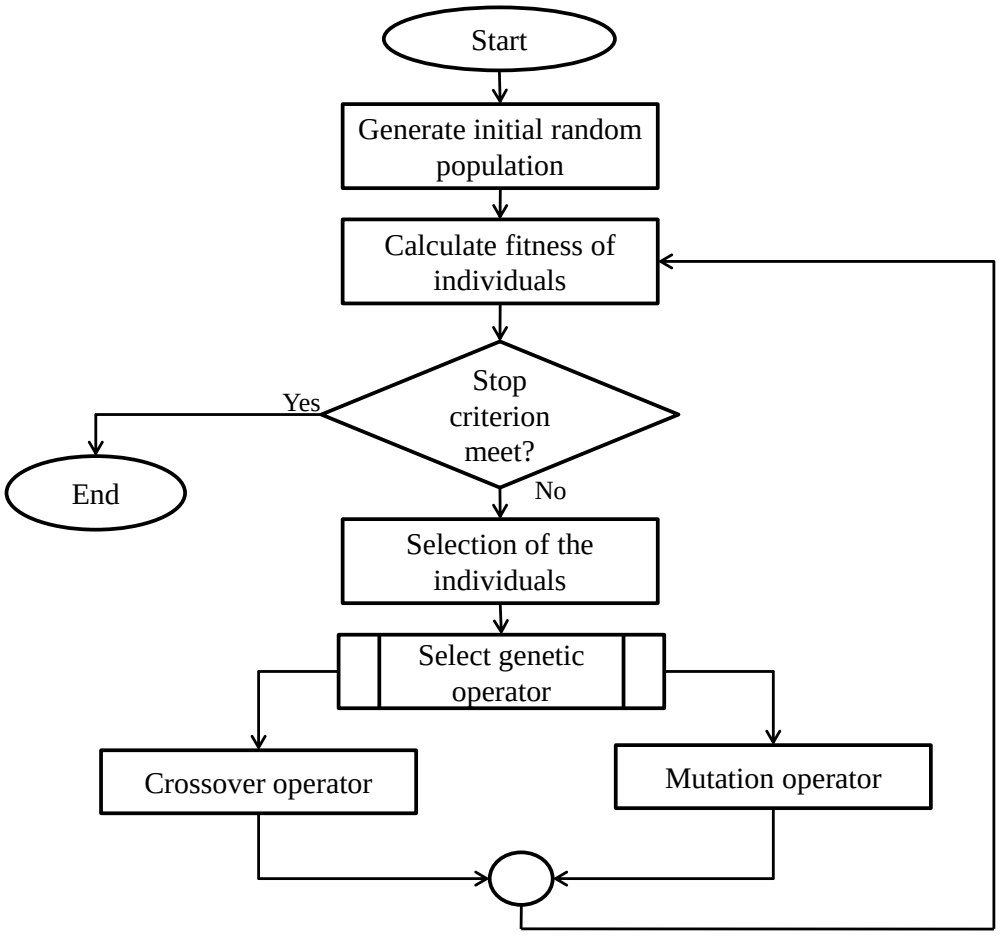



Figure 1.7: Flowchart of genetic algorithm

Figure 1.7 shows the flowchart of GA. Basically, GA consists of five main steps. These are initialization, selection, crossover, mutation, and stopping criteria. The working of GA is described as follows [47]:

i. **Population initialization:** A set of random chromosomes (*i.e.,* solutions) is generated. Each chromosome should be lying within the range of lower and upper bounds of the given problem.

ii. **Selection:** The value of fitness function is computed for each chromosome. The chromosome having highest fitness will be selected for preceding steps.

iii. **Crossover:** Two chromosomes are taken as parent $1$ and parent $2$. Thereafter, the crossover operation is applied on these chromosomes to produce two children. If any or both children have good fitness value than their respective parents, then it will survive. Otherwise, these are not selected for further evolution.

iv. **Mutation:** Every survived chromosome will be mutated. If mutated chromosome has better fitness than the previous one, it will survive.

v. **Stopping criteria:** If the stopping criteria is satisfied, then the algorithm will stop. Otherwise, Steps (ii-iv) are repeated. The well-known stopping criterion are number of iterations, number of function evaluations, or acceptance error. In this thesis, the acceptable error has been used to terminate the evolution of GA.

### 1.3.2 Differential evolution

Differential evolution is a well-known evolution based optimization technique [48]. It elegantly combines mutation and crossover operators of GA. Therefore, it has better convergence speed than the standard GA [49]. It consists of five main steps such as population initialization, mutation, recombination, selection, and stopping criteria. Figure 1.8 shows the flowchart of differential evolution. The steps of differential evolution are mentioned below:

i. **Population initialization:** Initially, random solutions are generated whose values lie between lower and upper bound. The normal distribution technique is mostly used to develop the random solutions.

ii. **Mutation:** In this, three distinct solutions are randomly selected for a target solution. A donor solution is generated by adding the difference of two solutions to third solution.

iii. **Recombination:** It is used to combine the successful solutions from the previous generation. It generates a trial solution from the elements of target solution and the donor solution of mutation step.

iv. **Selection:** To select the best solution, the fitness value of target solution is compared with trial solution. The fitness value of solutions depends upon the objective function. In case of maximization problem, if the fitness of trial solution is more than the target solution, then it is survived for next generation. The best known solution will be replaced with new solution.

v. **Stopping criteria:** If the termination criterion is satisfied, the algorithm stops. Otherwise, Steps ii-iv are repeated.

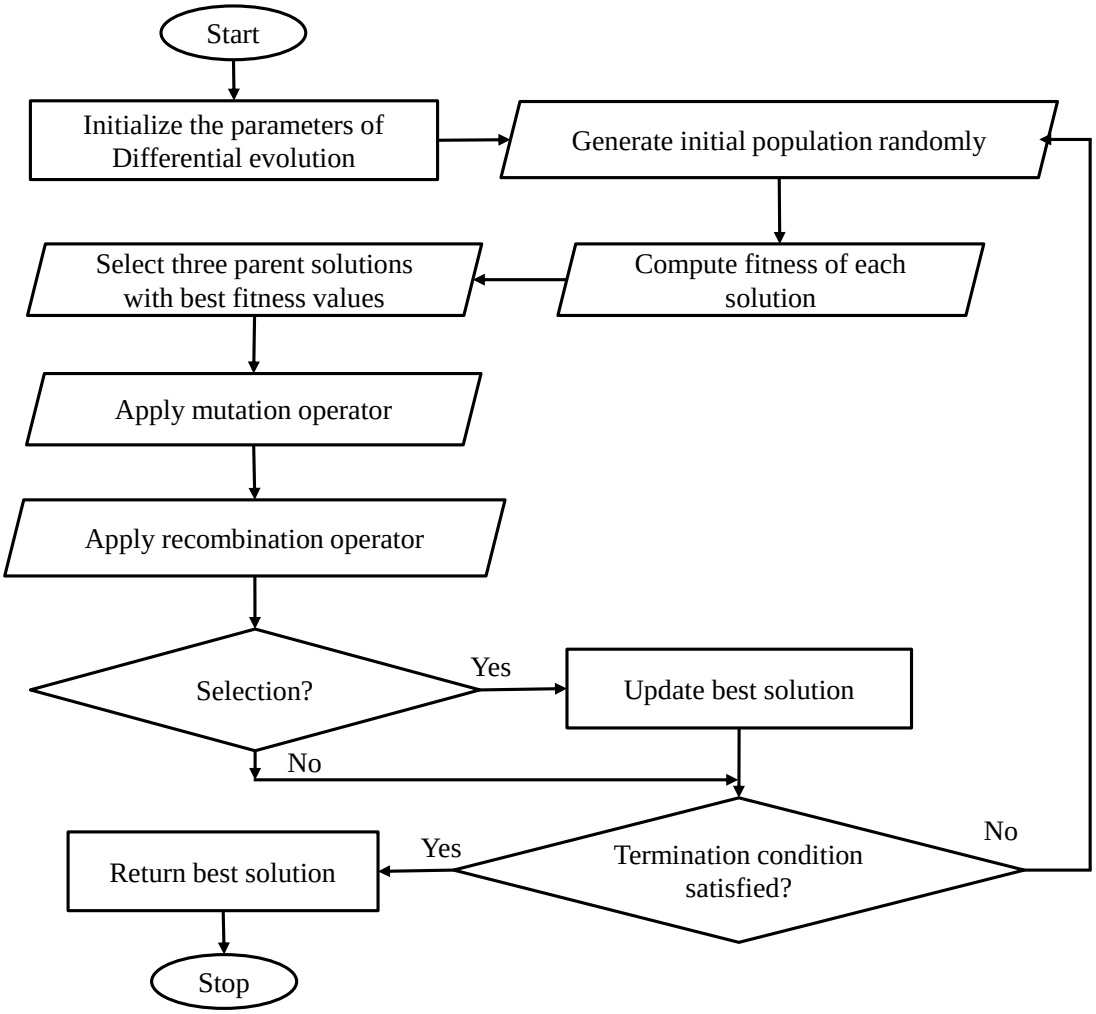



Figure 1.8: Flowchart of differential evolution

### 1.3.3 Non-dominated sorting genetic algorithm

The performance of Pareto frontier obtained from Non-dominated sorting genetic algorithm (NSGA-II) is mainly depend upon its initial parameters set [50]. NSGA-II requires tuning of input parameters. The selection of these parameters depends upon the user's experience in problem domain [50]. To overcome this issue, Reinforcement learning (RL) is

proposed by Bora *et al.* [51]. It is responsible for tuning the initial parameters of NSGA-II. It adjusts the variation of crossover and mutation operators during runtime by considering the results obtained from previous generation. In NSGA-II, RL is responsible for adjusting the operators namely probabilities of crossover and mutation.

### 1.3.4 Memetic differential evolution

Jia *et al.* [52] proposed a memetic differential evolution based on local chaotic search. It has two main advantages over standard differential evolution. First, it adaptively controls the parameters of differential evolution, *i.e.,* scaling factor ($F$) and Crossover rate ($CR$). It provides balance between fast convergence speed and population diversity. Second, differential evolution is well-known for global optimization, while it becomes stagnant in local search. Memetic differential evolution used chaotic system in local search to enhance the performance. Chaotic local search (CLS) is applied on refined solutions to save the execution time. Memetic differential evolution consists of following steps:

i. **Population initialization:** The initial population is generated using normal distribution. It contains a number of random solutions. The fitness of each solution of the population is computed.

ii. **Mutation:** Mutated solution is generated from three randomly selected solutions from the same population.

iii. **Crossover:** A trial solution is generated from the combination of best evaluated and mutated solutions based on the crossover rate.

iv. **Selection:** The fitness of trial solution is computed and compared with best evaluated solution. If trail solution has better fitness, then it will survive for the next generation. Otherwise, old solution will be continued for the next generation.

v. **Chaotic local search :** Local search is applied on best evaluated solution using a chaotic system to further refine the solution. It enhances the search capability of differential evolution and avoids the issue of premature convergence.

vi. **Termination condition:** If termination condition is met, then the algorithm stops. Otherwise, Steps (ii)-(v) are repeated.

### 1.3.5 Adaptive differential evolution

Lin *et al.* [53] proposed an Adaptive differential evolution (ADE). ADE enhances the population diversity and convergence speed by utilizing the immune algorithm. ADE is implemented by designing a novel selection operator and dynamic parameter control technique. It decomposes the initial population into dominated and non-dominated population. An

adaptive tuning technique is also designed to control the scaling factor ($F$) and Crossover rate ($CR$) to decrease the influence of parameter settings. $CR$ is regularly adapted with the evolutionary technique while $F$ is adaptively improved for each solution based on the success rate of offspring ($o_s$).

## 1.4 Performance measures

Performance evaluation measures are used to confirm the effectiveness of an image encryption technique against various security attacks.

### 1.4.1 Differential analysis

In differential attacks, attackers change the original image with single bit and encrypt the changed image using same security key. Thereafter, the attackers try to find the relation between encrypted images of both original and changed images [54]. Therefore, it is necessary to evaluate the sensitivity of the proposed technique towards small changes using differential analysis. The two well-known measures namely Number of pixel change rates (NPCR) and Unified average change intensity (UACI) are used to check the differential attacks.

#### 1.4.1.1 Number of pixel change rate

Number of pixel change rate (NPCR) is defined as the percentage of different pixel numbers between two encrypted images, whose plain images have only one pixel difference. If any technique provides high value of NPCR, then the algorithm is better against differential attacks [54]. NPCR can be computed as [54]:

$$NPCR = \frac{\sum_{i,j} D_{iff}(i,j)}{W \times H} \times 100 \tag{1.12}$$

Here,

$$D_{iff}(i,j) = \begin{cases} 0 & \text{if} \quad E(i,j) = E'(i,j) \\ 1 & \text{if} \quad E(i,j) \neq E'(i,j) \end{cases} \tag{1.13}$$

where $W$ and $H$ represent the width and height of the image, respectively. $D_{iff}(i,j)$ indicates the difference between corresponding pixels of encrypted image of the original image ($E(i,j)$) and encrypted image of changed image ($E'(i,j)$). The value of NPCR lies in the range of $[0, 100]$. NPCR value of an encrypted image should be close to $100$.

#### 1.4.1.2 Unified average changing intensity

Unified average changing intensity (UACI) measures the average intensity of difference between two encrypted images corresponding to plain images that have one pixel difference

[55]. It can be defined as [56]:

$$UACI = \frac{\sum_{i,j} E(i,j) - E'(i,j)}{255 \times W \times H} \times 100 \tag{1.14}$$

where $E(i,j)$ and $E'(i,j)$ are the pixel at $i^{th}$ row and $j^{th}$ column of encrypted images of original and changed images, respectively. The values of NPCR and UACI are to be maximized.

## 1.4.2 Statistical analysis

The encryption techniques can also be broken using statistical analysis of an encrypted image. Histogram analysis and correlation coefficient are used to analyze the adjacent pixels of an encrypted image to confirm the robustness of an encryption technique against statistical attacks.

### 1.4.2.1 Histogram analysis

Histogram analysis (HA) reveals the distribution of pixel values of an image. The histogram of original image should be totally different from the histogram of encrypted image. The histograms of plain images are non-uniform in nature. While the histograms of encrypted should be uniform in nature [5]. It means that all pixels are distributed equally in the space.

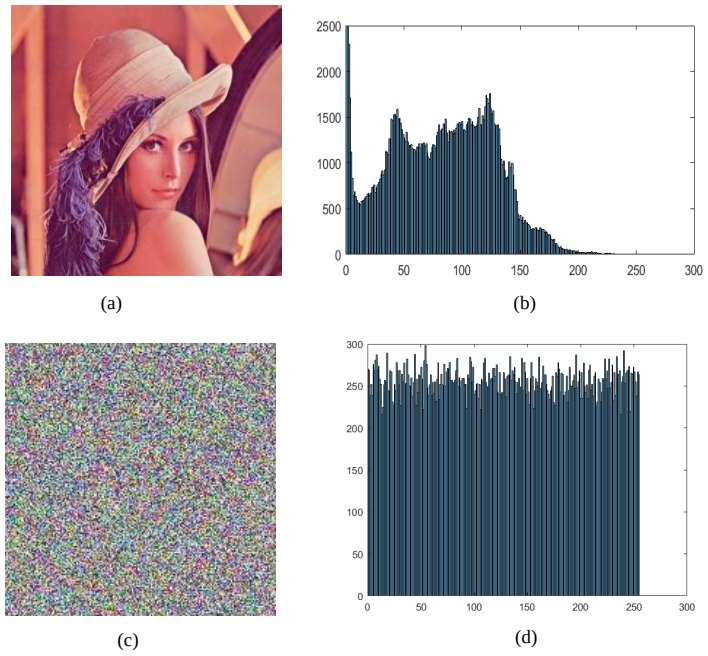



Figure 1.9: Histogram analysis (a) Plain Lena image, (b) Histogram of plain image, (c) Encrypted Lena image, and (d) Histogram of encrypted image

Figure 1.9 shows the histograms of plain and encrypted Lena images. It is observed from Figure 1.9 (b) that the histogram of plain Lena image is not uniform. From Figure 1.9 (d), it can be seen that the pixels of an encrypted image are uniformly distributed.

#### 1.4.2.2 Correlation coefficient

The values of adjacent pixels of an original image are strongly correlated in three directions, *i.e.,* horizontal, diagonal, and vertical. The good image encryption technique is one which reduces this relationship in ciphered image [57]. Correlation coefficient can be computed as [58]:

$$r_{x,y} = \frac{C(x,y)}{\sigma_x \sigma_y} \tag{1.15}$$

Here,

$$C(x,y) = \frac{\sum_{i=1}^{K}(x_i - E(x))(y_i - E(y))}{K} \tag{1.16}$$

$$\sigma_x = \sqrt{\frac{1}{K}\sum_{i=1}^{K}(x_i - E(x))^2} \tag{1.17}$$

$$\sigma_y = \sqrt{\frac{1}{K}\sum_{i=1}^{K}(y_i - E(y))^2} \tag{1.18}$$

where $C(x,y)$ is the covariance between samples $x$ and $y$. $K$ is the number of pixel pairs $(x_i, y_i)$. $\sigma_x$ and $\sigma_y$ are the standard deviation of $x$ and $y$, respectively. $E(x)$ and $E(y)$ represent the mean of $x_i$ and $y_i$, respectively. The value of $r_{x,y}$ lies in the range of $[-1, 1]$. The correlation value of an encrypted image should be near to $0$.

#### 1.4.2.3 Information entropy

Information entropy measures the average information per bit in an image. It contains the possible information available in the given image. Each pixel has different value. Therefore, the entropy of an encrypted image means each pixel has equal probability with uniform distribution [59]. It can be computed as:

$$H(S) = -\sum_{s}(P(s_i) \times log_2 P(s_i)) \tag{1.19}$$

where $H(S)$ represents the entropy of message source $(S)$. $P(s_i)$ denotes the probability of occurrence of $s_i$. The value of $H(S)$ lies in the range of $[0, 8]$. It should be close to $8$ for $8$-bit image [59].

### 1.4.3 Key analysis

Security keys are the core part of any encryption algorithm as the strength of the algorithm depends on it. The secret keys should be strong enough to resist all types of attacks. The desirable properties of a strong secret key are large key space and high sensitivity [60]. The key space depends on the size of secret key. If the size of secret key is large, then it is harder for an attacker to estimate the same key. Key sensitivity means if attacker modify even a single pixel in the original key, then the original image remains unrecoverable.

### 1.4.4 Noise attack

To destroy the useful information, an attacker may introduce noise in the encrypted image. Due to this, the intended user cannot recover the original image successfully after decryption. The attacker introduces additive, Gaussian, and Poisson noise in the encrypted image [5]. Therefore, an efficient image encryption technique should be resistant to noise attacks.

### 1.4.5 Mean squared error

Mean squared error (MSE) helps to compare the pixel values of the original image and decrypted image. The error is the amount by which the values of original image differ from decrypted image. MSE can be defined as [61]:

$$MSE = \frac{1}{WH} \sum_{i=1}^{i=W} \sum_{j=1}^{j=H} [I(i,j) - D(i,j)]^2 \tag{1.20}$$

where $i$ and $j$ are the pixel coordinates of images with size of $W \times H$ pixels. $I$ and $D$ are the original and decrypted images, respectively. The value of MSE lies in the range of $[0, \infty]$. The value of MSE between the original and decrypted images should be minimum.

### 1.4.6 Peak signal to noise ratio

Peak signal to noise ratio (PSNR) is used as a quality measurement between the original and decrypted images [62]. It can be mathematically computed as:

$$PSNR = 10log_{10} \frac{(2^n - 1)^2}{MSE} \tag{1.21}$$

where $n$ represents the number of bits per pixel. $PSNR$ is measured in decibel $(dB)$. The value of PSNR should be maximum between original and decrypted images. The value of PSNR lies in the range of $[0, \infty]$.

### 1.4.7 Mean absolute error

Mean absolute error (MAE) measures the difference between encrypted and original images [63]. It can be evaluated as:

$$MAE = \frac{1}{WH} \sum_{i=1}^{W} \sum_{j=1}^{H} |I(i,j) - E(i,j)| \tag{1.22}$$

where $I(i,j)$ and $E(i,j)$ denote original and encrypted images, respectively. $i$ and $j$ are the pixel coordinates of image with size of $W \times H$. The value of MAE lies in the range of $[0, 2^n - 1]$. Here, $n$ is the number of bits per pixel. It should be a maximum between original and encrypted images.

### 1.4.8 Signal to noise ratio

Signal to noise ratio (SNR) evaluates the results of an encryption algorithm quantitatively. It can be computed as [64]:

$$SNR = \frac{\sum_{i,j} [I(i,j)]^2}{\sum_{i,j} [I(i,j) - D(i,j)]^2} \tag{1.23}$$

where $I(i,j)$ and $D(i,j)$ represent original and decrypted images, respectively. The value of SNR lies in the range of [1,$\infty$]. The value of SNR should be maximum between original and decrypted images.

### 1.4.9 Execution time

Execution time (ET) is measured as the time required to execute a given image encryption technique. It is the aggregation of compile and run time. For the practical implementation, ET should be minimum. It is measured in milliseconds, seconds or minutes [65].

## 1.5 Research motivation

The strength of any image encryption technique depends upon the security key [66]. Therefore, many researchers have proposed different key generation techniques to improve the performance of image encryption techniques. Chaotic maps are widely used in image encryption techniques to develop the secret keys [67, 32]. The complex chaotic systems generate seemingly random sequences but actually deterministic behaviors [68]. Chaotic orbits are extremely sensitive to initial conditions and system parameters. These are very difficult or even impossible to predict in a long duration because a small difference in the initial conditions will lead to significantly different outcomes [69]. These properties of chaotic maps are used to design secure encryption techniques.

The first chaotic map was used in encryption by Matthews [70] in 1989. The chaos based image encryption techniques involve two main steps namely, permutation and diffusion. The permutation is used to change the location of pixels, while diffusion changes the pixels value using chaotic map [71].

Usama *et al.* [72] used multiple chaotic maps such as Chebyshev, Sine, Cubic, Tent, Henon, and Logistic to encrypt the images. The main drawback of this technique is that the initial parameters have been taken manually to generate a secret key. Therefore, it can be easily broken by cryptanalyst as reported in [73]. Zhu *et al.* [74] implemented an image encryption technique that performs permutation and diffusion at bit-level using Arnold transform and logistic map. However, the permutation based image encryption techniques are prone to known-plaintext and known-ciphertext attacks [75, 76].

Zhang *et al.* [77] utilized Discrete wavelet transform (DWT) with Piecewise linear chaotic map (PWLCM) to encrypt the images. MD5 has also been used to generate the initial values for chaotic maps. However, MD5 suffers from chosen-prefix collision attacks [78]. Wang *et al.* [79] used Deoxyribonucleic Acid (DNA) sequence and coupled map lattice to encrypt the images. The initial conditions for coupled map lattice are generated through extended hamming distance. The same key can be used to encrypt more than one image. However, this technique is not sensitive to the input image. Zhang *et al.* [80] proposed an image encryption technique based on $3$-D bit matrix. This technique overcomes the issues related to single bit based permutation. However, Wu *et al.* [81] broke this technique and proved that it cannot be used for secure communication. Because the initial conditions of secret keys did not depend on the plain image.

Kumar *et al.* [82] used elliptic curve cryptography with DNA to encrypt images. Akhavan *et al.* [83] proved that [82] is just a shuffling technique and does not provide significant confusion and diffusion. The original image can be easily recovered by applying only two chosen plain images. Zhou *et al.* [84] implemented an image encryption technique based on chaos line map. It provides better security than the existing techniques. Chen *et al.* [85] used a codebook attack on [84] and recovered the original image.

It can be analyzed from the above-mentioned facts that the security of encryption technique depends on the secret keys. The strength of secret keys lies in the generation of initial conditions of chaotic maps. The main motivation behind this research is to select the initial conditions in such a way that no one can easily recover the keys and original image using known-plaintext and chosen-plaintext attacks. The goal of our research is to propose the image encryption techniques based on evolutionary algorithms which select the initial parameters of chaotic maps according to the given plain image.

## 1.6 Thesis organization

Thesis consists of eight chapters and begins with fundamental concepts of image encryption. Subsequent chapters of the thesis have been organized in the following manner:

- **Chapter 2: Literature review**

  This chapter presents a systematic review of the state of art in the field of image encryption, with a special emphasis on the spatial and transform domain based encryption techniques. The comparison between existing image encryption techniques is also done to evaluate the gaps in the literature. The objectives and the thesis contributions are also discussed at the end of this chapter.

- **Chapter 3: Genetic based image encryption**

  An image encryption technique (*i.e.,* IGN) based on GA, beta chaotic map, and Non-subsampled contourlet transform (NSCT) is presented in this chapter. NSCT is used to decompose the input image into sub-bands. The initial parameters of beta chaotic map are optimized using GA. Thereafter, beta chaotic map generates a secret key using optimized parameters. Then, the secret key is utilized to encrypt the coefficients of sub-bands. Finally, the encrypted sub-bands are combined to produce the encrypted image. To test the security of IGN, the various experiments have been carried out such as statistical attack, differential attack, secret key, noise attack, and occlusion attack analyses. The experimental results have shown that IGN has better performance as compared to the existing image encryption techniques.

- **Chapter 4: Differential evolution based image encryption**

  In this chapter, an image encryption technique based on differential evolution (IDN) is proposed. In IDN, Arnold transform is utilized to permute the pixel's position of an input image. Differential evolution is used to tune the parameters required by a beta chaotic map. The entropy of an encrypted image is used as a fitness function. From the experimental results, the mean improvement of IDN has been observed over the others in terms of entropy, NPCR, UACI, PSNR and MAE are $0.22$ %, $0.09$ %, $0.10$ %, $20.5$ % (dB), and $9.9$ %, respectively. The mean reduction of IDN in case of a correlation coefficient is $2.9$ %. Therefore, the results reveal that IDN outperforms the others.

- **Chapter 5: Memetic differential evolution based image encryption**

  This chapter discusses the color image encryption technique based on memetic differential evolution (IIMA). In this technique, the initial parameters of the intertwining logistic map are tuned using memetic differential evolution. The input image is divided into red, green, and blue channels. The pixels of channels are permuted using Arnold transform. The intertwining map generates the secret keys to encrypt the permuted channels. In the end, the encrypted color channels are concatenated to obtain the encrypted image. The effectiveness of IIMA has been tested on five well-known color images. From results, the mean improvement has been observed in IIMA over the other techniques. The parameters of IIMA such as entropy, NPCR,

UACI, and PSNR have been improved by 0.051 %, 0.065 %, 0.097 %, and 2.17 % (dB), respectively. The correlation coefficient of IIMA has reduced by 0.8 %. The results reveal that IIMA provides higher efficiency and security as compared to the existing techniques.

- **Chapter 6: Parallel adaptive differential evolution based image encryption**

  An image encryption technique based on the Secure hash algorithm (SHA-3) and Adaptive differential evolution (ADE), namely ISAL, is proposed in this chapter. In this technique, ADE is used to optimize the input parameters of the Lorenz chaotic system. SHA-3 is used to generate a secret key based on the input image. The optimized parameters and external secret keys are used to generate the initial values for the Lorenz chaotic system. A parallel implementation of the proposed adaptive differential evolution based Lorenz chaotic system is also presented. The different types of security analyses such as statistical, differential attack, noise attack, and image enhancement, have been performed on ISAL. The comparative results reveal that ISAL provides better encryption results than the others. ISAL has the ability to resist against statistical, differential, noise, and image enhancement attacks.

- **Chapter 7:Non-dominated sorting genetic algorithm based image encryption**

  In this chapter, an image encryption technique that utilizes Fourier-Mellin moments and intertwining logistic map (IFIM) is discussed. Multi-objective Non-Dominated sorting genetic algorithm (NSGA-II) based on Reinforcement Llearning (MNSGA-RL) is used to optimize the required parameters of intertwining logistic map. Thereafter, permutation and diffusion operations are carried out on the input image using secret keys. IFIM is also implemented in a parallel fashion using master-slave architecture to increase the computational speed. The performance of IFIM is tested over five images and compared with seven well-known techniques. Experimental results have shown that IFIM provides encryption and decryption results at good computational speed as compared to the existing meta-heuristic based image encryption techniques.

- **Chapter 8: Conclusions and future work**

  This chapter presents the concluding remarks on the thesis by highlighting the significant contributions of the work done. The future directions of the implemented research work are also presented at the end of this chapter.

# Chapter 2

# Literature review

In this chapter, a comprehensive study of the existing well-known image encryption techniques is presented. These techniques can be divided into two main domains such as spatial and transform. Most of the image encryption techniques based on the combination of these domains. The spatial and transform domain based image encryption techniques are described in the preceding sections. The comparison between these image encryption techniques is also carried out using some well-known performance metrics. The research gaps are also identified with respect to various security attacks and performance metrics. Finally, the objectives of this research work are also discussed.

## 2.1 Spatial domain based image encryption techniques

The term spatial domain refers to the image plane itself which is direct manipulation of pixels. Spatial domain based techniques are those techniques which are directly applied on these pixels. The well-known spatial domain based image encryption techniques are chaotic, Deoxyribonucleic acid (DNA), cellular automata, meta-heuristic, and elliptic curve. The various spatial domain based techniques are discussed in subsequent subsections.

### 2.1.1 Chaos based image encryption techniques

Chaotic maps provide a promising alternative to traditional cryptosystems because of their properties such as deterministic nature, sensitive to initial values, ergodicity, and random behavior [86]. The inherent properties of chaotic maps are directly related to cryptographic characteristics of confusion and diffusion [87]. Thus, chaotic maps have been widely used in image encryption techniques. These are broadly classified into two categories such as one-dimensional and multi-dimensional. One-dimensional (1-D) systems are simple and have a high level of efficiency. But, these suffer from small key space and

weaker security. Multidimensional systems are complicated. But, these provide better key space and security against various attacks.

Pareek *et al.* [25] used two logistic maps and one external key to encrypt an image. The initial conditions of logistic maps are derived from the external key. The secret key is modified every time after encrypting the block of sixteen pixels of an image. Therefore, it is difficult for an attacker to discover the secret key. However, it is not sensitive to the input images.

Behnia *et al.* [88] combined (1-D) chaotic map with coupled map lattice to encrypt an image. This combination provides a large key space and high-level security. However, this technique has less sensitivity towards the input image.

Gao *et al.* [22] implemented a hyper-chaotic map in image encryption technique to reduce the prediction time than the simple chaotic maps based image encryption techniques. In this technique, shuffling of matrix is utilized to permute the pixels of an input image. Thereafter, hyper-chaotic map is used to diffuse the pixel values of shuffled image. This technique provides better key space and high security.

Ye *et al.* [89] used the logistic map to encrypt the images. In this technique, permutation is performed at bit level. This technique is robust against various security attacks. However, this technique suffers from known-plaintext and chosen-plaintext attacks [90].

Zhu *et al.* [74] proposed an image encryption technique that performs permutation and diffusion at bit-level. In this technique, Arnold transform and logistic map are utilized to perform permutation and diffusion operations. This technique is computationally efficient than the other image encryption techniques. However, permutation based image encryption techniques are prone to known-plaintext and known-ciphertext attacks [76, 75].

Mirzaei *et al.* [91] implemented an image encryption technique in parallel fashion. In this technique, an image is sub-divided into four blocks. Thereafter, chaotic map is used to shuffle these blocks. Then, each block is encrypted in the parallel scheme by changing the pixel values of each block. This technique has good computational speed.

Kanso *et al.* [92] utilized three-dimensional (3-D) Arnold transform to generate secret keys to encrypt the images. This technique is divided into three phases such as shuffling, scrambling, and masking. This technique is highly sensitive towards an input image. It provides better security against known-plaintext and chosen-plaintext attacks. Wang *et al.* [93] designed an image encryption technique that performs selective encryption using spatial chaotic maps. It chooses first 4-bits of each pixel for encryption. It proved that encrypting only $50$ % pixels of an input image provides better encryption results.

Khan and Shah [64] implemented S-box in a chaotic manner using affine and Lorenz transforms to improve the algebraic and statistical properties of S-box. S-box is mainly used in block encryption to create confusion among pixels. It provides better confusion and diffusion to protect the input image against security attacks.

Zhang and Wang [94] proposed an image encryption technique based on mixed linear-nonlinear coupled map lattices. The mixed linear and nonlinear coupled map lattices

overcome the issues of large periodic windows in bifurcations and small range of parameters. These maps are used to permute and diffuse an input image at bit level to generate an encrypted image.

Wang *et al.* [79] discussed an image encryption technique based on random integer cycle shift and $1$-D chaotic map. In this technique, cycle shift technology is used to change the pixel values of an input image. It is used in image encryption because of its easy implementation and can be extended to any size of image.

Wu *et al.* [95] presented an image encryption technique based on Chaotic system and DWT (CDWT). In this technique, secret key depends on both chaotic system and plain image. It further enhances the security of proposed image encryption technique.

Belazi *et al.* [54] improved scrambling technique using Permutation-substitution arrangement and chaotic techniques (PSCT). This technique has better security analysis and good key space. It consists of cryptographic phases such as diffusion, substitution, and permutation. These phases are carried out by enhanced chaotic map, S-box, logistic map, and permutation function to enhance the performance.

Liu *et al.* [96] improved the key space of [97] using a two-dimensional ($2$-D) sine map and an iterative chaotic map with infinite collapse modulation map ($2$-D-SIMM). The permutation and diffusion processes are combined into one step to reduce the computational time.

Ahmad *et al.* [3] designed an image encryption technique based on Orthogonal matrix and skew tent map (OSTM). It is designed to overcome the problems of [98]. OSTM provides better security against differential and statistical attack analysis.

Chen *et al.* [69] designed a four-dimensional ($4$-D) discrete chaotic map using sine function with an one-line equilibrium to encrypt the images. The key space generated by chaotic map is greater than $2^{1170}$. It provides a significant avalanche effect as compared to other image encryption techniques.

Hua *et al.* [99] used the concept of image filtering in an image encryption to enhance security. The image filtering can spread little change of plain-images to entire pixels of cipher-images. This technique provides better results than the other techniques in terms of statistical and differential attacks.

Pak and Huang [100] developed a new chaotic system using the difference of output sequences of two same chaotic systems which removes the drawbacks of a single chaotic map. However, the strength of image encryption technique is not so effective against known-plaintext attacks .

Li *et al.* [101] discussed a permutation and diffusion architecture in which permutation is performed at pixel level as well as the bit level to encrypt an image. This technique used five-dimensional ($5$-D )chaotic map to overcome the drawbacks of low dimensional chaotic maps.

Table 2.1 shows the comparison of chaos based image encryption techniques based on performance metrics. These metrics are NPCR, UACI, Key analysis (KA), HA, $r_{x,y}$, $H(S)$,

and Noise attack (NA) (discussed in Section 1.4). From the table, it can be observed that not even a single technique fulfills all the performance criteria. ✓ and ✗ represent given technique has considered and not considered the corresponding metric, respectively.

Table 2.1: Comparison of chaos based image encryption techniques

| Ref. | Technique | NPCR | UACI | KA | HA | $r_{x,y}$ | $H(S)$ | NA |
|---|---|---|---|---|---|---|---|---|
| [22] | Hyper-chaotic map | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ |
| [25] | Logistic map | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ |
| [35] | Chaotic algorithm | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ |
| [69] | Discrete chaotic map | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [74] | Arnold transform and Logistic map | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [79] | Integer cycle shift | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [54] | S-box and logistic map | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [88] | Coupled map lattice | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [91] | Chaotic map | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ |
| [92] | Arnold transform | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [93] | Spatial chaotic map | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [94] | Coupled map | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [95] | Chaotic map and DWT | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [96] | Sine map | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [100] | 1-D chaotic map | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| [101] | 5-D chaotic map | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [102] | Spatiotemporal chaos | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [103] | 3-D chaotic map | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ |
| [104] | Hyper-chaotic map | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ |

### 2.1.2 DNA based image encryption techniques

In recent times, researchers have designed a simulated environment of biological experiments on Deoxyribonucleic acid (DNA) technology named as pseudo-DNA technology. This idea has been promoted the development of DNA in the field of encryption [105].

Figure 2.1 shows the block diagram of DNA based image encryption process. Initially, the image is decomposed into Red (R), Green (G), and Blue (B) color channels. These three channels are transferred into binary matrices. DNA encoding rules are then applied to encode these matrices. To scramble the similarity between pixel values, DNA operations are applied to the encoded matrices. The decoding rules are then applied to convert them again into binary matrices. Finally, three color channels are combined to attain a cipher colored image [106].

The main reason for using DNA in image encryption are massive parallelism, ultra-low power consumption, and huge storage [79].

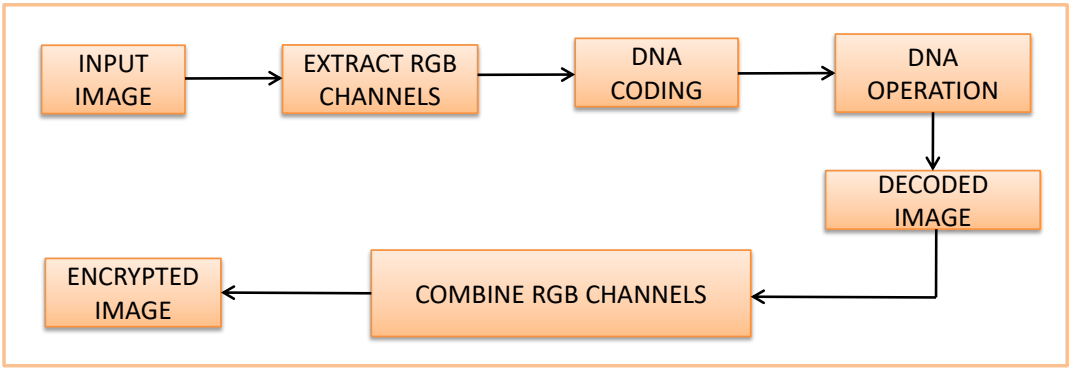



Figure 2.1: Block diagram of DNA based encryption technique

Wu *et al.* [107] proposed a color Image encryption technique using 1-D chaotic map and DNA (ICD). The input image and key stream are converted into matrices using DNA encoding rule. Thereafter, XOR and complementary operations are applied to scramble the matrices. The scrambled matrices are divided into equal blocks and shuffle them randomly. DNA addition and XOR operations performed on these matrices to get an encrypted image. This technique resists against chosen and known-plaintext attacks because three chaotic maps are used to generate a key stream that depends on both input image and secret keys.

Wang *et al.* [79] proposed an image encryption algorithm based on DNA sequence and coupled map lattice (DCML). The initial conditions for coupled map lattice are generated through extended hamming distance. The same key can be used to encrypt more than one image. Initially, an XOR operation is performed on the pixels of the input image by using coupled map lattice. DNA encoding rules are used to encode the confused image to obtain a DNA matrix. Furthermore, the shuffled DNA matrix is diffused using coupled map lattice. The final encrypted image is attained by decoding DNA rule. However, this technique is not sensitive towards the input image.

Kumar *et al.* [82] used a combination of DNA encoding rules and Elliptic curve Diffie-Hellman (DECDHE) for efficient image encryption. First, the colored image is encoded using DNA encoding rules to obtain confused RGB matrices and then DNA operations are applied on these matrices. Finally, asymmetric ECDHE cryptography is applied on image for encryption. The main benefit of this technique is to provide perfect forward secrecy and generate same security with small key size.

Li *et al.* [106] presented an improved image encryption technique for color images using DNA and real chaotic maps (DRCM). The quaternary coding is used in DNA sequence to improve the encoding efficiency and enhance the security of keys by using complex and real chaotic maps. The hamming distance is used to generate an one-time pad which further scramble the input image. This technique enhances the encoding efficiency of DNA based image encryption techniques.

Wang *et al.* [108] developed an Image encryption technique using DNA matrix, spatiotemporal chaos, and hamming distance (IDSH). Spatiotemporal chaos system is used to generate the secret keys. The hamming distance is utilized to permute DNA matrix. This technique has an ability to encrypt large size color images.

Mondal and Mandal [109] discussed a light-weight image encryption technique by using a Pseudo random number and DNA (PDNA). Two level encryption strategy is utilized. The encrypted technique has the ability to handle any kind of attack. However, the noise attack analysis has not been done in this technique.

Chai *et al.* [55] utilized Complex chaotic systems and DNA rules (CCD) to encrypt the plain images . Initially, color image is encoded using DNA rules. Thereafter, the complex chaotic map is used to encrypt the encoded image. This technique is very effective against noise and known-plaintext attacks.

Table 2.2 shows the comparison of DNA based image encryption techniques. DNA based image encryption techniques satisfy almost all of the performance metrics (can be seen from Table 2.2). However, these techniques have poor computational speed as compared to chaos based image encryption techniques.

Table 2.2: Comparison between DNA based image encryption techniques

| Ref. | Technique | NPCR | UACI | KA | HA | $r_{x,y}$ | $H(S)$ | NA |
|---|---|---|---|---|---|---|---|---|
| [79] | DCML | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [82] | DECDHE | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ |
| [55] | CCD | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| [106] | DRCM | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ |
| [107] | ICD | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| [108] | IDSH | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [109] | PDNA | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |

### 2.1.3 Cellular automata based image encryption techniques

Cellular automata (CA) consists of cells, which reside on grid with different forms of structures. These structures evolve through some finite time steps according to the specified rules based on states of neighboring cells. Therefore, CA simulates the complex structures. The huge amount of CA regulations allows several techniques to develop sequences. The reversible CA is extensively utilized by developers to execute block encryption technique. The main benefits of CA in encryption are huge amount of rules space and parallelism [110]. With the use of CA, image encryption techniques become lossless and adaptive for real-time applications. It also provides high security and fast operation [111]. Figure 2.2 shows the image encryption using cellular automata.

Wang and Luan [112] proposed a technique for image encryption using the combination of Chaotic map and reversible CA (CRCA). The confusion stage uses the chaotic map

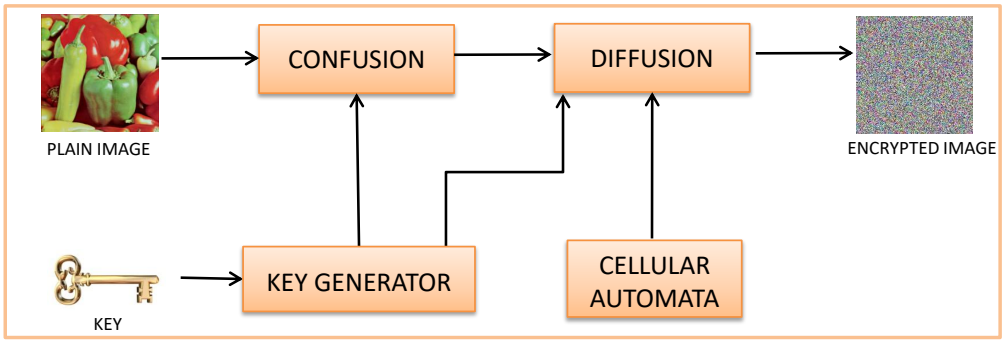



Figure 2.2: Image encryption using cellular automata

to generate key stream which permutes the bits of an image. In this technique, an image is divided into units and each unit contains $4$ bits. The diffusion stage considers only higher $4$ bits of each pixel because these higher bits provide all information about an image. The diffusion is achieved by applying reversible CA.

Bakhshandeh and Eslami [113] developed a new technique for Image encryption based on chaotic maps, CA, and permutation-diffusion architecture (ICCP). A piecewise linear chaotic map is used to permute the image in permutation phase. The diffusion phase uses the logistic map and reversible memory CA to diffuse the permuted image to obtain a secure image. This technique has an authentication ability.

Ping *et al.* [114] proposed an image encryption technique in which diffusion and confusion processes are done with the help of CA. CA generates good random sequences to create a scrambled image. Diffusion process uses the interaction between the local cells, whereas confusion process is achieved through CA rules by applying on these cells.

Li *et al.* [115] proved that the integral imaging based encryption procedures have an ability to implement secure and strong image cryptography. It focuses on depth-adaptation Integral images and hybrid cellular automata (IHCA). Initially, the actual image is divided into an elementary array by utilizing the depth-transformed integral technique. The evaluated elemental images are scrambled by utilizing HCA based chaotic maps. The superiority of this technique is its depth-transformed characteristic which consequently minimizes the magnification factor.

Mohamed [116] used CA to realize parallelization in image cryptography. Reversible CA (RCA) is used to construct pseudo-random permutation which is used on different blocks independently. The plain image is decomposed into blocks. Thereafter, the secret keys and nonce are applied to encrypt each block independently. This technique is computationally fast as compared to other techniques.

Enayatifar *et al.* [117] introduced a hybrid model for image encryption which contains Chaotic maps, DNA, and CA (CDCA). The input image is encrypted by using the sequences and operators of DNA with the help of CA. The rule number is selected through the chaotic map. It suffers from poor computational speed.

Yang *et al.* [118] utilized a Quantum CA (QCA) in image encryption. The primary benefit of this technique is that it has a time complexity of $O(n)$ which is less than the traditional quantum encryption technique $O(n^2)$.

Li *et al.* [119] designed an image encryption technique that uses depth-conversion Integral imaging and CA (ICA) to enhance the security. Depth-converted integral imaging technique is used to decompose the input image into an elemental image array. Thereafter, CA and chaotic map is utilized to encrypt the image. This technique reduces the magnification factor that degrades the reconstruction process.

Chai *et al.* [58] employed Memristive hyper-chaotic system, CA, and DNA sequence operations (*i.e.,* MCD) for image encryption. SHA-$256$ hash function is used to generate the secret key and compute the initial values of chaotic system. Moreover, a dynamic DNA encoding scheme is introduced. This technique has an ability to resist known-plaintext and noise attacks.

YaghoutiNiyat *et al.* [120] proposed a Non-uniform CA (NUCA) framework for image encryption to solve the problems associated with limited number of reversal rules. The key image is created using non-uniform CA and then hyper-chaotic mapping is used to select random numbers for encryption.

The image encryption techniques based on $2$-D CA masks provide encryption results with horizontal patterns. To overcome this issue, Li *et al.* [111] proposed an image encryption algorithm based on CA. In this technique, CA pixel-permutation (CAPP) is used to break the established orders of pixels. It provides large key space and highly sensitive towards secret keys.

The comparison between CA based image encryption techniques has been depicted in Table 2.3. Most of the techniques provide better results in terms of performance measures. The speed of these techniques is better as compared to DNA based image encryption techniques.

Table 2.3: Comparison of cellular automata based image encryption techniques

| Ref. | Technique | NPCR | UACI | KA | HA | $r_{x,y}$ | $H(S)$ | NA |
|---|---|---|---|---|---|---|---|---|
| [58] | MCD | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| [111] | CAPP | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✓ |
| [112] | CRCA | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [113] | ICCP | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [114] | CA | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [115] | IHCA | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |
| [116] | RCA | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [117] | CDCA | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [118] | QCA | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| [119] | ICA | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ |
| [120] | NUCA | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

### 2.1.4 Meta-heuristics based image encryption techniques

Meta-heuristics techniques are used in image encryption to optimize the parameters required by chaotic maps to generate the secret keys.

Abdullah *et al.* [47] used genetic algorithm (GA) for image encryption. This technique is used to select best encrypted image from the initial population. The chaotic technique is utilized to develop a given number of encrypted images. Thereafter, GA is used to select the best encrypted image which has high entropy and low correlation coefficient.

Sreelaja *et al.* [121] developed a technique to create secure keys by utilizing Ant colony optimization (ACO) algorithm. It reduces the burden of storage and distribution of keys. It enhances the security by encoding keys using a code table. In this technique, plain image is represented in the form of characters and key-stream is generated from the combinations of plaintext characters.

Enayatifar *et al.* [122] used weighted discrete imperialist competitive algorithm (WDICA) in image encryption. It provides best optimization results with maximum entropy and minimum correlation coefficients. The chaotic map is used to generate an initial population for algorithm and then weighted discrete ICA is used to select the encrypted image with highest entropy and low correlation coefficient.

Enayatifar *et al.* [46] proposed an image cryptosystem based on hybrid GA and DNA sequence, so called GDNA. It has an ability to improve the quality and choose the best optimum mask. DNA sequence and chaotic map are used to generate DNA masks. Then, GA is used to select the optimum DNA mask. GA is used to maximize entropy during the encryption.

Abbas [123] described an image encryption algorithm based on Independent component analysis (ICoA) and Arnold cat map (ACM). ACM is used to create a random mixing matrix by injecting an arbitrary image. The source images are converted into vectors and combined with mixing matrix to generate the encryption images. This technique can encrypt more than one image with the same process. Finally, ICoA is used to decrypt the images.

Talarposhti and Jamei [124] proposed an encryption algorithm for gray scale images based on Dynamic harmony search (DHS). This technique uses the skew tent map to create encrypted images. DHS selects the best encrypted image and then performs diffusion and permutation to increase the entropy and minimize correlation coefficient. The main drawback of this technique is its computation time.

Table 2.4 shows the performance comparison of meta-heuristic based image encryption techniques. The main drawback of these techniques is computational speed. However, these techniques provide better encryption as compared to the existing image encryption techniques.

Table 2.4: Comparison of meta-heuristic based image encryption techniques

| Ref. | Technique | NPCR | UACI | KA | HA | $r_{x,y}$ | $H(S)$ | NA |
|---|---|---|---|---|---|---|---|---|
| [46] | GDNA | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [47] | GA | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [121] | ACO | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ |
| [122] | WDICA | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [123] | ICoA | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| [124] | DHS | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |

### 2.1.5 Elliptic curve and fuzzy based image encryption techniques

Elliptic curves are based on the properties of algebraic curves. Koblitz and Miller [66] developed a public key encryption technique by utilizing elliptic curve. The main features of elliptic curve encryption are small key size and better computational [66].

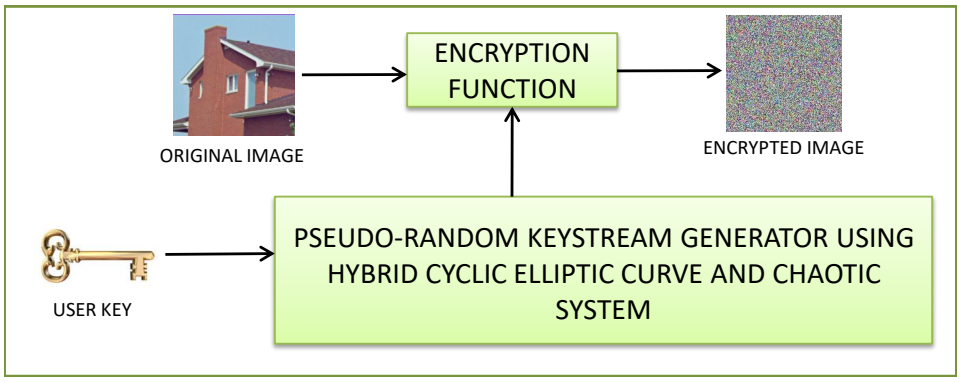



Figure 2.3: Encryption process using elliptic curve and chaotic map

Figure 2.3 shows the role of an elliptic curve in image encryption. A pseudo-random keystream developed using the cyclic elliptic curve and chaotic technique. It contains two phases such as allocation for development of initial keystream using a chaotic map and merging with pseudo-random bit sequence [65]. An image is initially decomposed into a binary data sequence. Then, data is masked with random keystream. The generation of keystream is done through hybridization of elliptic curve and chaotic system. The analogous encrypted image is attained.

El-Latif and Niu [65] proposed a technique to develop strong keystream based on hybridization of Elliptic curve and chaotic maps (ECC). The input image is converted into a data stream. Then, secret keys are applied on encryption function which is generated from the combination of elliptic curve and chaotic technique to mask the data. The encrypted data stream is converted into pixels of an image to get the ciphered image.

Behnia *et al.* [125] introduced an image encryption technique based on Jacobian elliptic maps (JEM). These maps are used to remove the drawbacks of chaotic cryptosystems such as small key space and weak security.

Nagaraj *et al.* [126] proposed a new image encryption technique which is the combination of Elliptic curve cryptography and magic matrix operations (ECCMM). The input image is embedded on points of the elliptic curve using the transform algorithm. The image is decomposed into data matrices and each pixel of an image is represented by a magic matrix. Thereafter, each pixel is encoded using elliptic cryptography function to generate an encrypted image.

Liu *et al.* [127] introduced a new color image encryption technique based on Choquet fuzzy integral (CFI) and chaotic map (CFICM). This technique uses a piecewise linear chaotic map to generate the secret keys and Lorenz map. It is used to initialize the inputs of CFI. CFI creates a random key-streams which are used to confuse and diffuse an image to obtain a decrypted image.

Seyedzadeh *et al.* [128] used CFI to generate a keystream to encrypt the color images. This technique has three phases such as generation of keystream, circular shift, and diffusion process. CFI is used to generate the pseudo-random key-streams and bits of each color pixel are shifted circularly based on key-stream. The permuted bits are encrypted using the combination of keystream and color pixels.

Table 2.5 shows the performance comparison of elliptic and fuzzy based image encryption techniques. Both elliptic and fuzzy techniques are mainly used in image encryption to generate efficient secret keys.

Table 2.5: Comparison of elliptic and fuzzy based image encryption techniques

| Ref. | Technique | NPCR | UACI | KA | HA | $r_{x,y}$ | $H(S)$ | NA |
|---|---|---|---|---|---|---|---|---|
| [65] | ECC | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [125] | JEM | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [126] | ECCMM | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| [127] | CFICM | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [128] | CFI | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ |

## 2.2 Transform based image encryption techniques

Transform domain based image encryption techniques have been extensively used in the field of image encryption. The given image is transformed from spatial to frequency domain by using a suitable transform model. Figure 2.4 shows the working of transform domain based image encryption technique. It uses the double transform to encrypt the image [129]. For color images, the majority of techniques decompose the input color image into three color channels (*i.e.,* R, G, and B channels). Each color channel is converted into

a transform domain for the encryption process.

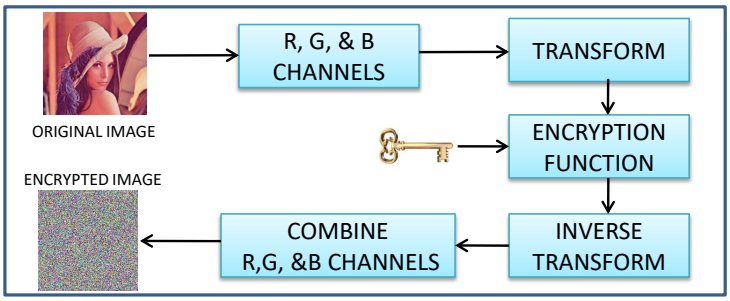



Figure 2.4: Generic framework of transform based image encryption technique

The well-known transforms are fractional Mellin transform, Fractional Fourier transform (FrFT), Gyrator transform (GT), Discrete cosine transform (DCT), affine transform, *etc*. These techniques are discussed in the preceding sections.

### 2.2.1 Gyrator transform based image encryption techniques

Singh and Sinha [130] implemented a novel cryptosystem using chaos in gyrator domain. In this technique, a scrambled image is developed by utilizing the gyrator domain and dual chaotic masks. The tent, Kaplan-Yorke, and logistic map are used to encrypt the image. However, the computational speed of this technique is low.

Wang *et al.* [131] used linear exchanging operation and random phase encoding in Gyrator transform (GT) domain for double image encryption. In the linear exchanging operation, two primitive images are linearly recombined *via* a random orthogonal transform matrix. The resultant blended images are employed to constitute a complex-valued image. Then, the image is encoded into a noise-like encrypted image by a Double phase random encoding (DRPE). This technique is highly sensitive towards the fractional orders of GT.

Wang *et al.* [132] used a Modified gerchberg-saxton algorithm (MGSA) with Phase-only mask (POM) in GT domain for encryption. It reduces the crosstalk effect on multiplexing images.

Abuturab [133] proposed an asymmetric image encryption technique for color images. A color image is decomposed into three channels and each channel is altered using Hartley transform. The altered channels are combined to get the first scrambled image. The first decryption key is generated by truncating the phase and amplitude of altered channels. GT is applied on scrambled image to get the final encrypted image and second decryption key. The decryption process is a reversible process to retrieve the original image.

Chen *et al.* [134] presented the solution for cross-talk disturbance found in phase based images. This technique implements double image encryption using GT and local pixel scrambling scheme. Two images are encoded into a complex function and then this

function is shuffled using local pixel scrambling. The shuffled image is rotated using GT to enhance the cryptosystem.

Yao *et al.* [135] encrypted the color images with the help of deduced GT. The primary feature of this algorithm is its decryption process. The encryption process involves gyrator and Fourier transform. But, decryption process involves only inverse of Fourier transform. This technique enhances the security of image encryption technique against various attacks.

Table 2.6 shows the performance comparison of GT based image encryption techniques. It can be concluded from the Table 2.6 that gyrator based image encryption techniques do not satisfy all the required performance measures. These techniques do not provide information regarding differential analysis.

Table 2.6: Comparison of gyrator transform based image encryption techniques

| Ref. | Technique | NPCR | UACI | KA | HA | $r_{x,y}$ | $H(S)$ | NA |
|---|---|---|---|---|---|---|---|---|
| [61] | Gyrator wavelet transform | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ |
| [130] | GT and chaos | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ |
| [131] | GT and DRPE | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✓ |
| [132] | MSGA with POM | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| [133] | GT and Hartley transform | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ |
| [134] | Pixel scrambling in GT | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| [135] | Deduced GT | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ |

### 2.2.2 Fractional Fourier transform based image encryption techniques

Wang *et al.* [136] proposed an optimistic technique of optical image scrambling by utilizing binary Fourier transform. The keys are developed using an order of scrambled pixels. This technique provides good security and robust against noise and distortion attacks. The main advantage of this technique is easy to implement.

Guo *et al.* [137] designed an improved color image cryptosystem by utilizing Arnold transform and discrete fractional random transform. R, G, and B channels of an input color image are converted into Intensity-hue-saturation (IHS) color space. The intensity channels are encrypted by using discrete fractional random transform and Arnold transform. This technique saves the storage space of cryptosystem keys. Due to the transform domain, it loses the potential detail in the actual image which degrades its performance.

Li *et al.* [138] proposed an encryption technique to encrypt multiple images using cascaded fractional Fourier transform (FrFt). The original images are decomposed into two phase masks. One phase mask is used to generate keys and another phase mask is used to encrypt the images. The decryption process uses reverse FrFt to obtain the original image and decryption keys are different from encryption keys which are obtained from a different group of a mask.

Li and Lee [139] tried to overcome the drawback of occlusion in double image encryption using Modified computational integral imaging reconstruction (MCIIR). FrFT technique is used to encrypt the elemental image array which is stored through pickup process. The reconstruction process uses MCIIR to obtain actual images which enhance the resolution of recovered images. The disadvantage of this technique is that it increases transmission overhead in a network.

Ran *et al.* [140] suggested a solution to solve the problem of information-independence in image encryption by applying Non-separable fractional Fourier transform (NFrFT). This technique has a potential of tangling the information along and across the directions together, which is not possible in FrFT and GT.

### 2.2.3 Fresnel, wavelet and cosine transform based image encryption techniques

Zhao *et al.* [141] proposed a multiple-image encryption technique based on the Position multiplexing of Fresnel domain (PMFD). This technique is less time-consuming because of its non-iterative nature. The encryption key can be further designed to realize a better reconstruction of plaintext.

Wang *et al.* [142] suggested the solution of silhouette problem which is presented in interface based encoding techniques using Fresnel transform and random phase modulation (FTRPM). This technique used a single beam implementation. Therefore, there is no need of beam splitting during the decryption process. It is also a time-saving technique because of non-iterative nature.

Wang *et al.* [143] addressed the issue of cross-talk noise in multiple-image encryption techniques. They implemented Retrieval algorithm and phase mask multiplexing in Fresnel domain (RPFD). In this, each image is encrypted individually into a phase-only function to remove the noise.

Wang *et al.* [144] tried to eliminate the threat of information disclosure in image cryptosystems based on Phase-truncation technique (PTS). For this, Random amplitude mask (RPM) is used to remove the information disclosure risk. This technique can be extended to other domains such as gyrator, Fourier, and Fractional Fourier.

Luo *et al.* [145] suggested an encryption architecture using Integer wavelet transform (IWT). In this architecture, the decomposition process is done through IWT to divide the input image into approximation and detail coefficients. The approximation coefficients are diffused using spatiotemporal chaos. Then, the diffused image is obtained by inverse IWT. The permutation process is performed to reduce correlation among pixels using a logistic map to get an encrypted image.

Mehra and Nishchal [61] used the combination of GT and Wavelet transform (WT) to protect the phase images. The secret key is created by different random phase codes as well as parameters of GT and WT. Therefore, the developed secret key is stronger than the

earlier techniques.

Kanso and Ghebleh [146] contributed in the field of visual image encryption by implementing an embedded process in Lift wavelet transform (LWT). It enhances the security of encryption techniques and quality of the resultant images.

Lima *et al.* [147] suggested the use of Cosine number transform (CNT) for medical images encryption. The technique is used to avoid round-off errors and maintain a high quality of images. It divides the image into blocks. Then, each block is sequentially applied to CNT. The encrypted image is obtained when the whole image is processed.

Wu *et al.* [148] used Reality-preserving fractional discrete cosine transform (RPFrDCT) to encrypt the color images. The encrypted image of this technique is a single color image. Therefore, it is convenient for storage and transmission.

Yaru and Jianhua [149] proposed an image encryption algorithm based on FrDCT *via* Polynomial interpolation (PI-FrDCT), and Dependent scrambling and diffusion (DSD) process. The sinusoidal chaotic map is used to generate the pseudo-random sequence which is utilized by PI-FrDCT to encrypt the images. The coefficients of PI-FrDCT are also limited by sigmoid function. DSD is applied to generate an encrypted image.

Table 2.7 shows the comparison of Fourier, Fresnel, wavelet, and cosine transforms based image encryption techniques. Transform based image encryption techniques have significant encryption speed. Table 2.8 summarizes the pros and cons of well-known image encryption techniques.

Table 2.7: Comparison between transform based image encryption techniques

| Ref. | Technique | NPCR | UACI | KA | HA | $r_{x,y}$ | $H(S)$ | NA |
|---|---|---|---|---|---|---|---|---|
| [136] | Fourier transform | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| [137] | Fractional transform | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| [138] | Cascaded FrFT | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ |
| [139] | FrFT and MCIIR | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| [140] | NFrFT | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✓ |
| [141] | PMFD | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ |
| [142] | FTRPM | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ |
| [143] | RPFD | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ |
| [144] | RPM | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ |
| [145] | IWT | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [146] | LWT | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✓ |
| [147] | CNT | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| [148] | DCT | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ |
| [149] | PI-FrDCT and DSD | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ |

Table 2.8: Pros and cons of encryption techniques

| Technique | Pros | Cons |
|---|---|---|
| Chaos | i. Deterministic behavior<br>ii. Unpredictable and nonlinear. | i. Suffers from small key space and weak security.<br>ii. Poor computational speed. |
| DNA | i. High computational speed. ii. Minimum storage requirement. iii. Energy efficient. | i. Not suitable for digital computing environment. |
| CA | i. Easy to implement.<br>ii. High degree of security.<br>iii. Run in parallel manner. | i. Keys must be kept secret. |
| Meta-heuristics | i. Better quality of decrypted image<br>ii. Provide high degree of security. | i. Premature convergence<br>ii. Poor convergence speed. |
| GT | i. Passive, linear, and lossless technique.<br>ii. Non-reciprocal transform.<br>iii. Does not modify the range of data. | i. Not suitable for non-stationary signals. |
| FFT | i. Suitable for spectral analysis.<br>ii. Able to capture non-repetitive events.<br>iii. Able to store the waveforms. | i. It provides only the frequency information of an image.<br>ii. Not preferable for linear and high order polynomial shapes. |
| WT | i. Best suitable for non-stationary signal analysis.<br>ii. Temporal information retained in transform process. | i. Computationally intensive.<br>ii. Less efficient and natural.<br>iii. Wavelets take more energy to implement itself correctly. |
| DCT | i. Orthogonal transform based upon compression.<br>ii. Better computation good speed. | i. May introduce random noise due to quantization.<br>ii. Blocking artifacts. |

Continued on next page

**Table 2.8 – continued from previous page**

| Technique | Pros | Cons |
|---|---|---|
| Elliptic Curve | i. Equal level of security even with small key size.<br>ii. Very fast key generation. | i. High computation time.<br>ii. Require secure random generator. |
| Fuzzy | i. Suitable for uncertain and approximate reasoning.<br>ii. Generate initial conditions for chaotic maps.<br>iii. Sensitive to initial conditions. | i. Rules must be known in prior.<br>ii. Require extensive computation.<br>iii. Difficult to implement. |

## 2.3 Cryptanalysis on image encryption techniques

In cryptanalysis, the cryptanalyst knows everything about cryptosystem except secret key. Cryptanalyst launch different types of attacks on cryptosystem to explore relation between plaintext and ciphertext to recover the secret key. The different types of attacks are ciphertext only, known plaintext, chosen plaintext, and chosen ciphertext. The various secure image encryption techniques have been broken by cryptanalysts. They suggest further improvements in encryption technique to enhance their security.

Wang and He [150] cryptanalysed Zhang and Liu [151] image encryption technique using chosen-plaintext attack. It was observed that the plain image can be retrieved without knowing the secret key. An image encryption technique that uses both skew tent and hyper chaotic maps proposed by Kadir *et al.* [27]. This technique is cryptanalysed by [73]. The complete keystream can be retrieved through chosen-plaintext attack. Murillo *et al.* [152] implemented an image encryption technique that was dependent on the plain image. However, this technique was cracked by Fan *et al.* [153] using known/chosen plaintext attacks.

Zeng *et al.* [154] broke Li *et al.* [155] image encryption technique by launching chosen-plaintext and known-plaintext attacks. The security of broken technique can be improved through dynamic permutation process.

Su *et al.* [156] revealed that the image cryptosystem designed by [157] using chaos and DNA is susceptible to chosen-plaintext attack. They suggested that it can be improved by replacing the entropy present in chaos system with hash function.

Zhang *et al.* [158] used chosen-plaintext attack to break the encryption algorithm proposed by [159]. To improve the security, they proposed a modification in keystream of [159]. Norouzi and Mirzakuchaki [160] revealed that Zhao *et al.* [161] image encryption technique suffers from weak secret key. The keystream of [161] does not depend on the plain-image. Therefore, the secret key can be easily recovered using the chosen-plaintext

attack.

It can be observed from the cryptanalysis that chaotic map based image encryption techniques can be broken easily. There is need to replace the chaotic map with another technique which can provide more security. So, it is still a challenging issue to make chaotic map more secure.

## 2.4 Research gaps

After a detailed analysis of literature review, the following research gaps have been formulated.

i. Majority of the existing image encryption techniques use some constants to accomplish image encryption process. These constants limit the performance of image encryption algorithms because the given constant value(s) may be effective for some set of images, but not for all images. Hence, finding the optimistic parameter values for to design an efficient encryption technique is still an ill-posed problem.

ii. Selecting an appropriate objective function is a difficult task. Hence, the evolutionary optimization techniques are not widely used in earlier work in the field of image encryption.

iii. Parallel processing is not used in most of the image encryption. The size of image is becoming huge nowadays such as medical and remotely sensed images. Therefore, it becomes essential to use parallel processing in image encryption.

iv. Majority of the meta-heuristic techniques suffer from poor computation speed as they have poor convergence speed. Therefore, selection of an appropriate evolutionary technique for image encryption techniques is still an open research area of research.

## 2.5 Objectives

Based on the research gaps mentioned in Section 2.4, the objectives of this research work are as follows:

i. To study the performance of evolutionary based image encryption techniques.

ii. To design and develop efficient color image encryption techniques using evolutionary optimization techniques.

iii. To implement proposed image encryption techniques in the parallel processing environment to enhance their efficiency.

iv. To compare the proposed techniques with existing image encryption techniques using well-known quality metrics.

# Chapter 3

# Genetic based image encryption

## 3.1 Introduction

Chaotic maps are widely used in the field of image encryption due to their random behavior. The chaotic maps require initial parameters to generate the secret keys. The assignment to these parameters is mainly done by manually or trial and error technique. The poor assignment can greatly affect the performance of an image encryption technique. To address this issue, an Image encryption using Genetic algorithm in Nonsubsampled contourlet transform (IGN) is developed in this chapter.

## 3.2 Nonsubsampled contourlet transform

Nonsubsampled contourlet transform (NSCT) is an extension of Contourlet transform (CT) that having shift-invariant feature [162].

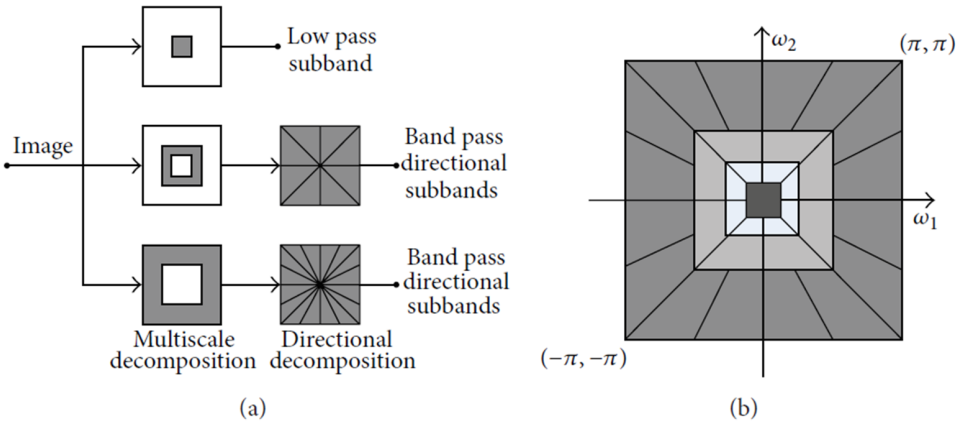



Figure 3.1: Nonsubsampled contourlet transform (a) NSFB structure that implements NSCT and (b) Idealized frequency partitioning obtained using NSFB structure

Figure 3.1 (a) shows the Nonsubsampled filter bank (NSFB) structure that implements NSCT. The idealized frequency partitioning obtained using NSFB structure is depicted in Figure 3.1 (b). The multiscale decomposition feature of CT is achieved by using Laplacian pyramids (LPs). Directional filter banks (DFBs) is used to generate the directional decomposition of CT. LP and DFB utilize downsamplers and upsamplers, respectively [163]. Therefore, CT is not shift-invariant. NSCT is designed using Nonsubsampled pyramids (NSP) and nonsubsampled DFBs to achieve shift-invariant feature (see Figure 3.1) [164].

### 3.2.1 Nonsubsampled pyramid

Nonsubsampled pyramid (NSP) guarantees the multiscale characteristic of NSCT. NSP depends upon two-channel nonsubsampled 2-D filter banks. Figure 3.2 (a) shows the three stage decomposition of NSP. An efficient reconstruction condition can be defined as [164]:

$$H_0(z)G_0(z) + H_1(z)G_1(z) = 1 \tag{3.1}$$

Here, $G_0(z)$ and $G_1(z)$ represent low and high pass reconstruction filters, respectively. $H_0(z)$ and $H_1(z)$ represent low and high pass decomposition filters, respectively.

The perfect frequency support of low pass filter at $j^{th}$ level is the region $[-\frac{\pi}{2^j}, \frac{\pi}{2^j}]^2$. In the same way, the support of high pass filter is the complement of low pass support region on $[-\frac{\pi}{2^{j-1}}, \frac{\pi}{2^{j-1}}]^2 \setminus [-\frac{\pi}{2^j}, \frac{\pi}{2^j}]^2$. The equivalent filters of $j^{th}$ level cascading NSP can be defined as [164]:

$$H_n(z) = \begin{cases} H_1(z^{2^{n-1}}) \prod_{j=0}^{n-2} H_0\left(z^{2^j}\right), & 1 \leq n < 2^k \\ \prod_{j=0}^{n-2} H_0\left(z^{2^j}\right), & n = 2^k \end{cases} \tag{3.2}$$

Here, $H_0(z)$ and $H_1(z)$ represent low and high pass filter at initial level, respectively. $k$ and $n$ represent number of decomposition levels and total number of decompositions of NSCT, respectively. Figure 3.2 (b) shows the prefect frequency of NSP.

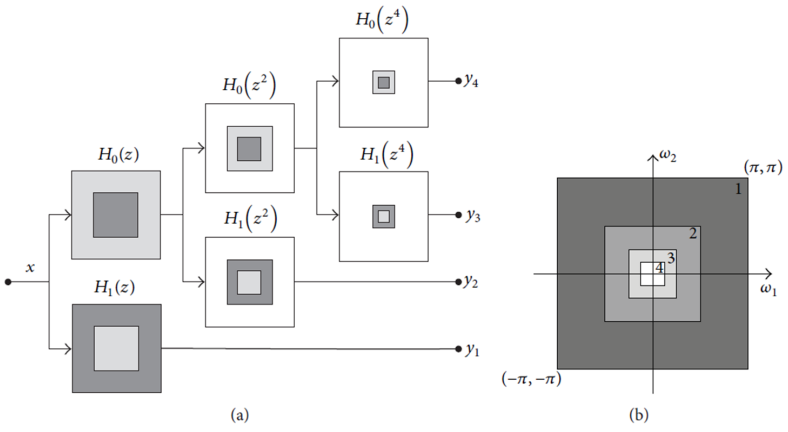



Figure 3.2: Nonsubsampled pyramid (a) Three-stage pyramid decomposition and (b) Frequency divisions of a nonsubsampled pyramid

### 3.2.2 Nonsubsampled directional filter bank

Nonsubsampled directional filter bank (NSDFB) is fully shift-invariant and divides the 2D-frequency plane into directional wedges. It is evaluated by discarding the downsamplers and upsamplers in DFB [165]. Figure 3.3 shows a four-channel directional decomposition. The equivalent filter in each channel is defined as [165]:

$$U_k(z) = U_i(z)U_j(z^Q) \tag{3.3}$$

Here, $U_k$, $U_i$, and $U_j$ represent upsampled fan filters, which have checker-board frequency support [162]. The values of $i$ and $j$ are either $0$ or $1$.

### 3.2.3 Combining NSP and NSDFB

NSCT is achieved by integrating NSP and NSDFB (see Figure 3.1 (a)). NSP contains multi-scale decomposition and captures the point discontinuities. NSDFB has directional decomposition feature. It links point discontinuities into linear structures [166]. NSDFB can be repeated continually on low pass sub-band obtained from NSP. Hence, NSCT is appropriate for image encryption as it provides shift-invariance, multidirection, and multiresolution.

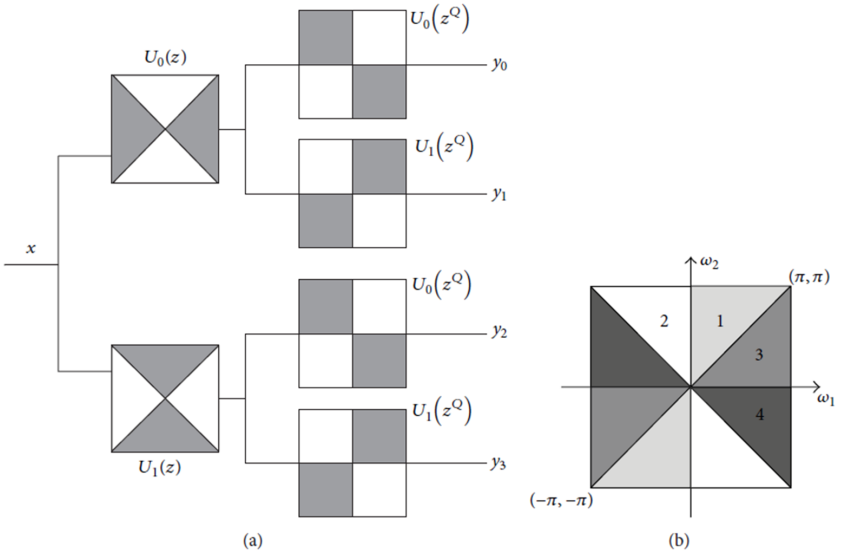



Figure 3.3: Four-channel NSDFB developed with two-channel fan filter bank (a) Filtering structure and (b) Corresponding frequency partitioning

## 3.3 Genetic algorithm based image encryption

The proposed Image encryption using GA in NSCT (IGN) consists of two processes. These are encryption and decryption.

### 3.3.1 Encryption process

Initially, NSCT is applied on the input image ($I$) of size $W \times H$. It decomposes $I$ into three sub-bands such as High-frequency sub-band 1 ($H^1_{w,h}$), High-frequency sub-band 2 ($H^2_{w,h}$), and Low-frequency sub-band ($H^0_{w,h}$). Thereafter, GA is used to optimize the required parameters of beta chaotic map. Beta chaotic map is used to encrypt and decrypt the coefficients of sub-bands. Inverse of NSCT is applied on encrypted sub-bands to produce the final encrypted image. Algorithm 1 describes the various steps involved in IGN.

**Algorithm 1:** Image encryption using genetic algorithm in NSCT domain

**Input:** Input image $I$

**Output:** Encrypted image $E$

Initialize random population of chromosomes ($c_s$) with size $C_l$.
**for** $j \leftarrow 1 : P_{op}$
    $c_s \leftarrow rand(C_l)$
    $\alpha_e \leftarrow c_s(C_l)$
    [$b_{en}$, $b_{sol}$, $Best_{en}$] $\leftarrow Fitness$ ($I$, $c_s$, $b_{en}$, $\alpha_e$)
**end**
*/* Repeat the following steps till the stopping criteria is not met. */*
**while** (max($Best_{en}$) $\leq A_e$) **do**
    */* Apply crossover operator. */*
        [$c_1$, $c_2$] $\leftarrow Crossover$ ($b_{sol}$, $C_r$)
        */* Now evaluate the fitness for both children i.e., $c_1$, $c_2$. */*
        $\alpha_e \leftarrow c_1(C_l)$
        [$b_{en}$, $b_{sol}$, $Best_{en}$] $\leftarrow Fitness$ ($I$, $c_1$, $b_{en}$, $\alpha_e$)
        $\alpha_e \leftarrow c_2(C_l)$
        [$b_{en}$, $b_{sol}$, $Best_{en}$] $\leftarrow Fitness$ ($I$, $c_2$, $b_{en}$, $\alpha_e$)
    */* Apply mutation operator to develop new solution. */*
        $new \leftarrow Mutation$ ($b_{sol}$, $M_r$)
        $\alpha_e \leftarrow new(C_l)$
        [$b_{en}$, $b_{sol}$, $Best_{en}$] $\leftarrow Fitness$ ($I$, $new$, $b_{en}$, $\alpha_e$)
**end**
Return best solution, *i.e.,* $b_{sol}$.
*/* Encrypt image using GA based optimized parameters. */*
$B_k \leftarrow BetaChaoticMap(W, H, b_{sol}(1 : (C_l - 1))$
$\alpha_e = b_{sol}(C_l)$
$EH^0_{w,h} \leftarrow \mod (B_k \times H^0_{w,h} + (1 - \alpha_e) \times B_k, p_k)$
$EH^1_{w,h} \leftarrow \mod (B_k \times H^1_{w,h} + (1 - \alpha_e) \times B_k, p_k)$
$EH^2_{w,h} \leftarrow \mod (B_k \times H^2_{w,h} + (1 - \alpha_e) \times B_k, p_k)$
$E \leftarrow iNSCT\big(\, EH^0_{w,h}, EH^1_{w,h}, EH^2_{w,h} \,\big)$

In Algorithm 1, $c_s$ lies between [0, 1]. $b_{en}$ and $Best_{en}$ represent best fitness and best fitness so far, respectively. $b_{sol}$ contains $c_s$ with good fitness values. The detailed description of steps mentioned in Algorithm 1 are given below:

#### 3.3.1.1 Secret key generation using beta chaotic map

Beta chaotic map is sensitive towards initial conditions and variation in bifurcation parameter. Hence, it enhances the security of encryption techniques against various security attacks. The key generation process based on beta chaotic map is described in Algorithm 2.

**Algorithm 2:** Secret key generation using beta chaotic map

**Input:** Row size $W$, column size $H$, and chromosome length $C_l$

**Output:** Secret key $B_k$

**Begin:** $B_k \leftarrow BetaChaoticMap(W, H, c_p(1 : (C_l - 1))$
/* $c_p(5)$, $c_p(6)$ *and* $c_p(9)$ *represent* $l_1$, $m_1$ *and* $b$ */
$u \leftarrow c_p(5) + c_p(6) \times c_p(9)$
/* $c_p(7)$ *and* $c_p(8)$ *represent* $l_2$ *and* $m_2$ */
$v \leftarrow c_p(7) + c_p(8) \times c_p(9)$
/* $c_p(3)$ *and* $c_p(2)$ *represent* $y_2$ *and* $y_1$ */
$y_c \leftarrow (u \times c_p(3) + v \times c_p(2))/(u + v)$
/* *Evaluate secret key* */
**for** $i \leftarrow 1 : W - 1$
  **for** $j \leftarrow 1 : H - 1$
    /* $c_p(1)$ *initialize the first element of beta chaotic map* */
    $y(1, 1) \leftarrow c_p(1)$
    **if** $y(i, j) > c_p(2)$ && $y(i, j) < c_p(3)$ **then**
      $t_1 \leftarrow \frac{y(i,j)-c_p(2)}{y_c-c_p(2)}$
      $l \leftarrow \frac{c_p(3)-y(i,j)}{c_p(3)-y_c}$
      $beta(i, j) \leftarrow t_1^u \times l^v$
    **else**
      $beta(i, j) \leftarrow 0$
    **end**
      /* $c_p(4)$ *represents* $t$ */
    $y(i + 1, j + 1) \leftarrow c_p(4) \times beta(i, j)$
  **end**
**end**
**return** $B_k \leftarrow y$

#### 3.3.1.2 Population initialization

GA is used to generate initial population randomly using Normal distribution ($ND$) with mean ($\mu$=0) and variance ($\sigma^2 = 1$). The size of every chromosome ($c_s$) is $10$. The values of lower and maximum bound are $0$ and $1$, respectively. The population size of GA is represented by $P_{op}$. The value of $P_{op}$ is set to $50$.

#### 3.3.1.3 Multi-objective fitness function and selection operator

The three performance measures such as entropy, Number of pixel change rates (NPCR), and Unified average change intensity values (UACI) are used in designing a multi-objective fitness function. The chromosome, which has maximum value of summation of these three measures, will be elected as a best chromosome. Algorithm 3 depicts the steps involved for selecting the best-fit chromosome.

**Algorithm 3:** Selection using multi-objective fitness function

**Input:** Input image $I$, chromosome $c_l$, best fitness $b_{en}$, and encryption factor $\alpha_e$

**Output:** $b_{en}$, best solution $b_{sol}$, fitness matrix $Best_{en}$

**Begin:** $[b_{en}, b_{sol}, Best_{en}] \leftarrow Fitness(I, c_s, b_{en}, \alpha_e)$

*/* Input image is decomposed by using NSCT into sub-bands. $p_k$ is the peak pixel value. $i$ represents the number of function evaluation. The initial values of $b_{en}$ and $i$ are $0$ and $1$, respectively. */*

$[H^1_{k,l}, H^2_{k,l}, L] \leftarrow NSCT(I)$

$[W, H] \leftarrow size(I)$

$B_k \leftarrow BetaChaoticMap(W, H, c_s(1 : (C_l - 1))$

$EH^0_{w,h} \leftarrow \mod (B_k \times H^0_{w,h} + (1 - \alpha_e) \times B_k, p_k)$

$EH^1_{w,h} \leftarrow \mod (B_k \times H^1_{w,h} + (1 - \alpha_e) \times B_k, p_k)$

$EH^2_{w,h} \leftarrow \mod (B_k \times H^2_{w,h} + (1 - \alpha_e) \times B_k, p_k)$

$E \leftarrow iNSCT\big(\, EH^0_{w,h}, EH^1_{k,l}, EH^2_{k,l} \,\big)$

$$c_{en} \leftarrow \left( \frac{entropy(E)}{8} + \frac{NPCR(E) + UACI(E)}{2} \right)$$

*/* Selection operator */*

**if** $b_{en} \leq c_{en}$ **then**

$b_{en} \leftarrow c_{en}$

$b_{sol}(i) \leftarrow c_s$

$Best_{en}(i) \leftarrow b_{en}$

$i \leftarrow i + 1$

**end**

The multi-objective fitness function can be defined as:

$$\begin{aligned} &\text{Maximize} \quad f(z) = \frac{H(S)}{8} + \left(\frac{NPCR + UACI}{2}\right) \\ &\text{subject to} \quad H(S) \geq t_h, \end{aligned} \tag{3.4}$$

where $H(S)$ represents entropy of an encrypted image ($E'$). $t_h$ denotes minimum required entropy value.

#### 3.3.1.4 Crossover operator

Crossover is used to develop new chromosomes from the existing chromosomes. It is used to swap the genes between two or more parent chromosomes and produces two or more new children. In this work, one point crossover operator is used to develop new chromosomes. The value of fitness function is computed after applying the crossover operator. If the evaluated fitness values of one or both children have more than their parents, then children will survive for future iterations. Otherwise, they will die. Algorithm 4 describes the steps required to perform the crossover operator.

**Algorithm 4:** Crossover operator

**Input:** Best solutions $b_{sol}$ and crossover rate $C_r$

**Output:** New children $c_1$ and $c_2$ from parents $p_1$ and $p_2$

**Begin:** $[c_1, c_2] \leftarrow Crossover\ (b_{sol}, C_r)$
**if** $rand \geq C_r$ **then**
  $[a1 \quad b1] \leftarrow size(b_{sol})$
  **for** $ii \leftarrow 2 : b1$
    $p_1 \leftarrow b_{sol}(ii - 1)$
    $p_2 \leftarrow b_{sol}(ii)$
    $c_1 \leftarrow p_1$
    $c_2 \leftarrow p_2$
    */* Swap the half of population to perform the crossover operation */*
    $c_1\Big((C_l \times C_p + 1) : C_l\Big) \leftarrow p_2\Big(C_l \times C_p + 1 : C_l\Big)$
    $c_2\Big(C_l \times C_p + 1 : C_l\Big) \leftarrow p_1\Big(C_l \times C_p + 1 : C_l\Big)$
  **end**
**end**
**return** $(c_1, c_2)$

#### 3.3.1.5 Mutation

Mutation is another well-known operator of GA. It alters the position of given solution either by interchanging the elements over these positions or just shuffle the given solution. After applying the mutation operator, it is just required to evaluate the fitness of mutated chromosome. If the mutated chromosome has more fitness than its parent, then it will survive for further. Otherwise, they will die. The entire procedure of mutation is depicted in Algorithm 5.

**Algorithm 5:** Mutation operator

**Input:** Best solutions $b_{sol}$ and mutation rate $M_r$

**Output:** New mutated solution $new$

**Begin:** $new \leftarrow Mutation\ (b_{sol}, M_r)$
**if** $rand \geq M_r$ **then**
  $[a1 \quad b1] \leftarrow size(b_{sol})$
  **for** $ii \leftarrow 1 : b1$
    $new \leftarrow b_{sol}(ii)$
    $m_{p1} \leftarrow rand_p\left(1, \frac{C_l}{2}\right)$
    $m_{p2} \leftarrow rand_p\left(\frac{C_l}{2} + 1, C_l\right)$
    $temp \leftarrow new(m_{p1})$
    $new(m_{p1}) \leftarrow new(m_{p2})$
    $new(m_{p2}) \leftarrow temp$
  **end**
**end**
**return** $new$

#### 3.3.1.6 Stopping criteria

In this work, the acceptance error ($A_e$) is used to stop the evolution of GA. GA will repeat all its steps till the best fitness value is less than $A_e$. Once any best fitness value greater or equal to $A_e$ is found, the algorithm returns optimized parameters for beta chaotic map. The stopping criteria can be represented as

$$A_e = \begin{cases} 1, & \text{if} \quad A_e \leq c_{en}, \\ 0, & \text{otherwise.} \end{cases} \tag{3.5}$$

### 3.3.2 Decryption process

The original image can be extracted from encrypted image using decryption process, *i.e.,* described in Algorithm 6. To decrypt the encrypted image, receiver needs correct security key $(B_k)$ and encryption parameter ($\alpha_e$).

**Algorithm 6:** Decryption process

**Input:** Encrypted image $E$, secret key $B_k$, and encryption factor $\alpha_e$

**Output:** Decrypted image $D$

**Begin:** $D \leftarrow Decryption(E, B_k, \alpha_e)$

*/* Encrypted image is decomposed into sub-bands */*

$[EH^1_{w,h}, EH^2_{w,h}, EH^0_{w,h}] \leftarrow NSCT(E)$

*/* Decryption of encrypted sub-bands */*

$DH^0_{w,h} \leftarrow (EH^0_{w,h} - (1-\alpha_e) \times B_k)/\alpha_e$

$DH^1_{w,h} \leftarrow (EH^1_{w,h} - (1-\alpha_e) \times B_k)/\alpha_e$

$DH^2_{w,h} \leftarrow (EH^2_{w,h} - (1-\alpha_e) \times B_k)/\alpha_e$

*/* Apply inverse NSCT of decrypted sub-bands to produce the decrypted image */*

$D \leftarrow iNSCT\big(DH^0_{w,h}, DH^1_{w,h}, DH^2_{w,h}\big)$

**return** $D$

## 3.4 Experimental results and discussion

The simulation results are carried out in MATLAB $2013a$ operating on $2.20\ GHz$ core $i5$ Processor with $8$ GB RAM on Windows $10$. The parameters setting of IGN is mentioned in Table 3.1. These parameters are set according to trial and error on small simulations.

Table 3.1: Parameters setting of genetic based image encryption

| Parameter | Value |
|---|---|
| Decomposition level ($k$) | $1$ |
| Chromosome length ($C_l$) | $10$ |
| Population size ($P_{op}$) | $50$ |
| Mutation rate ($M_r$) | $0.01$ |
| Crossover rate ($C_r$) | $0.2$ |
| Acceptance error ($A_e$) | $2^{-10}$ |
| Maximum iterations ($M_{it}$) | $50$ |
| Crossover point ($C_p$) | $0.5$ |
| Peak pixel value ($p_k$) | $256$ |

### 3.4.1 Images and techniques involved for comparison

IGN has been tested on ten images [167]. The first five gray images are Cameraman, Lena, Baboon, Pirate, and Woman. The next five color images are Boat, Airplane, Peppers, House, and Lake. The size of these images is $256 \times 256$. Five well-known meta-heuristic based image encryption techniques such as GA [47], ACO [121], WDICA [122], GDNA [46], and DHS [124] are used for comparison.

### 3.4.2 Visual analysis

Figures 3.4 and 3.5 show the results obtained on gray and color images using IGN, respectively. Note that only red channel results of color images are considered.

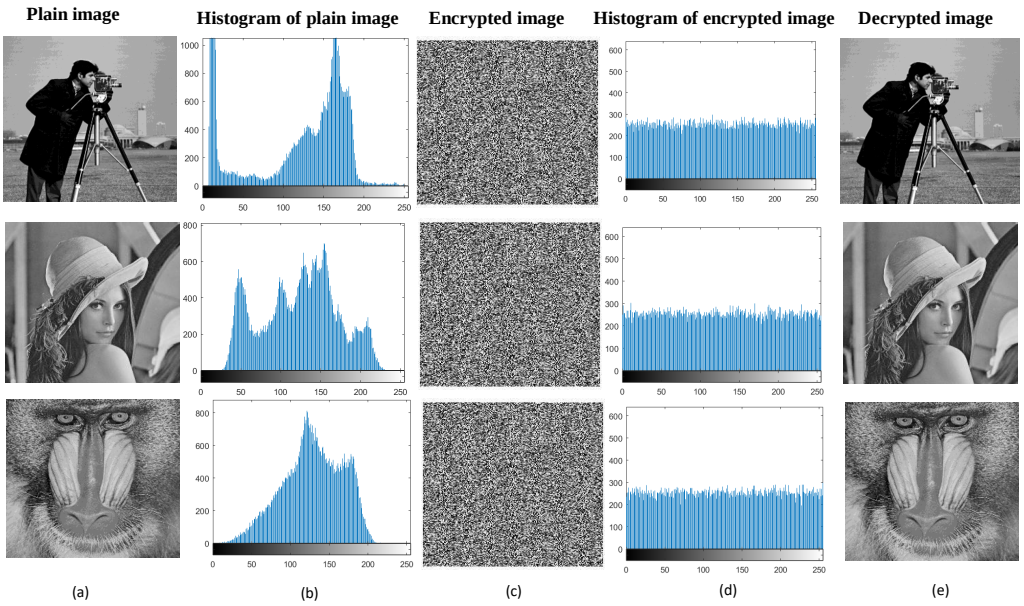



Figure 3.4: Visual analysis of IGN (a) Plain images, (b) Histogram of plain images, (c) Encrypted images, (d) Histogram of encrypted images, and (e) Decrypted images

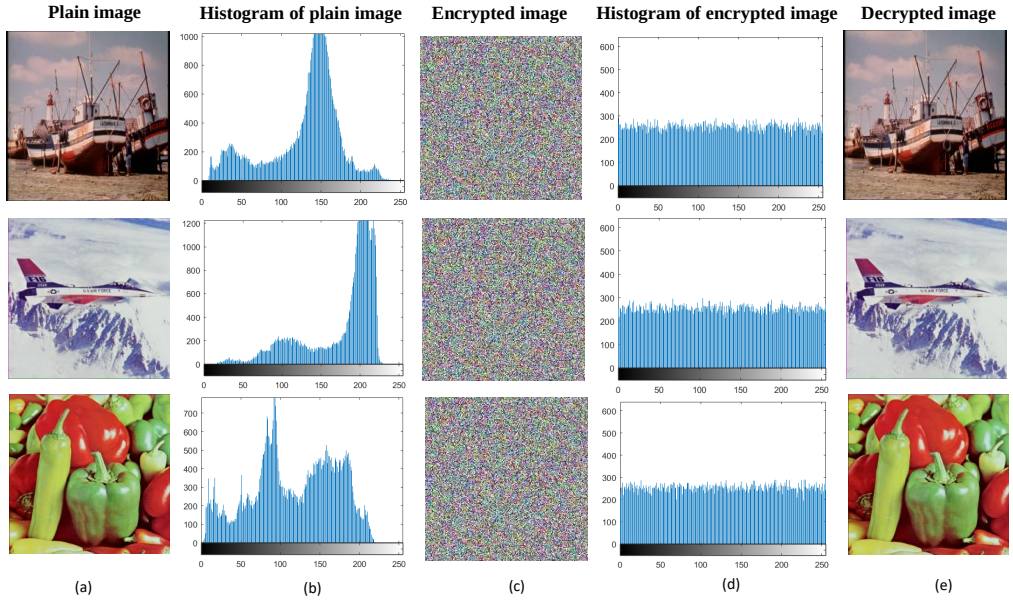



Figure 3.5: Visual analysis of IGN (a) Plain images, (b) Histogram of plain images, (c) Encrypted images, (d) Histogram of encrypted images, and (e) Decrypted images

### 3.4.3 Security analysis

In this section, the security analysis of IGN has been done. The five well-known security analysis namely statistical attack, differential attack, secret key, occlusion attack, and noise attack analyses have been used to test the robustness of IGN.

#### 3.4.3.1 Histogram analysis

Figures 3.4(b) and 3.5(b) show the histograms of gray plain images and color plain images, respectively. Figures 3.4(d) and 3.5(d) show the histograms of encrypted gray and color images, respectively. From Figures 3.4(d) and 3.5(d), it is observed that the pixels of encrypted images are uniformly distributed. Thus, it is hard to find out any information from the encrypted images.

#### 3.4.3.2 Correlation analysis

To investigate IGN, the horizontal, diagonal, and vertical correlation between adjacent pixels of input and encrypted image is computed. Table 3.2 depicts Horizontal (Hcorr), Vertical (Vcorr), and Diagonal (Dcorr) correlation coefficients of test images and their respective encrypted images. It has been observed from Table 3.2 that the attacker cannot find any relationship between adjacent pixels to break the algorithm.

Table 3.2: Correlation coefficient analysis of IGN

| **Images name** | **Plain image** | | | **Cipherd image** | | |
|---|---|---|---|---|---|---|
| | Hcorr | Vcorr | Dcorr | Hcorr | Vcorr | Dcorr |
| Cameraman | 0.9556 | 0.9738 | 0.934 | −0.0001 | 0.0036 | 0.0073 |
| Lena | 0.9258 | 0.9593 | 0.9037 | 0.0012 | −0.0063 | 0.0058 |
| Baboon | 0.8701 | 0.8411 | 0.7889 | 0.0001 | −0.0008 | 0.0002 |
| Pirate | 0.9434 | 0.9564 | 0.9134 | 0.0022 | 0.0006 | −0.0029 |
| Woman | 0.9914 | 0.9925 | 0.9859 | 0.0032 | 0.0028 | 0.0023 |
| Boat | 0.9269 | 0.9452 | 0.8834 | 0.0111 | 0.0010 | 0.0024 |
| Airplane | 0.9396 | 0.9332 | 0.8884 | 0.0045 | −0.0021 | −0.0021 |
| Peppers | 0.9675 | 0.973 | 0.9432 | −0.0015 | −0.0046 | −0.0041 |
| House | 0.9846 | 0.9813 | 0.9682 | −0.0036 | −0.0002 | 0.0044 |
| Lake | 0.9580 | 0.9577 | 0.9295 | −0.0003 | −0.0008 | −0.0076 |

Figure 3.6 (a) shows horizontal, vertical, and diagonal correlation analysis of plain cameraman's image. It can be seen that the adjacent pixels of a plain image are highly correlated with each other. Therefore, it may reveal the statistical information of an image. Figure 3.6 (b) shows the horizontal, vertical, and diagonal correlation analysis of an encrypted cameraman's image. From figure, it can be observed that pixels are seen random

in the space. Which implies that there is no relation among the adjacent pixels. Hence, attacker cannot extract any statistical information from an encrypted image.

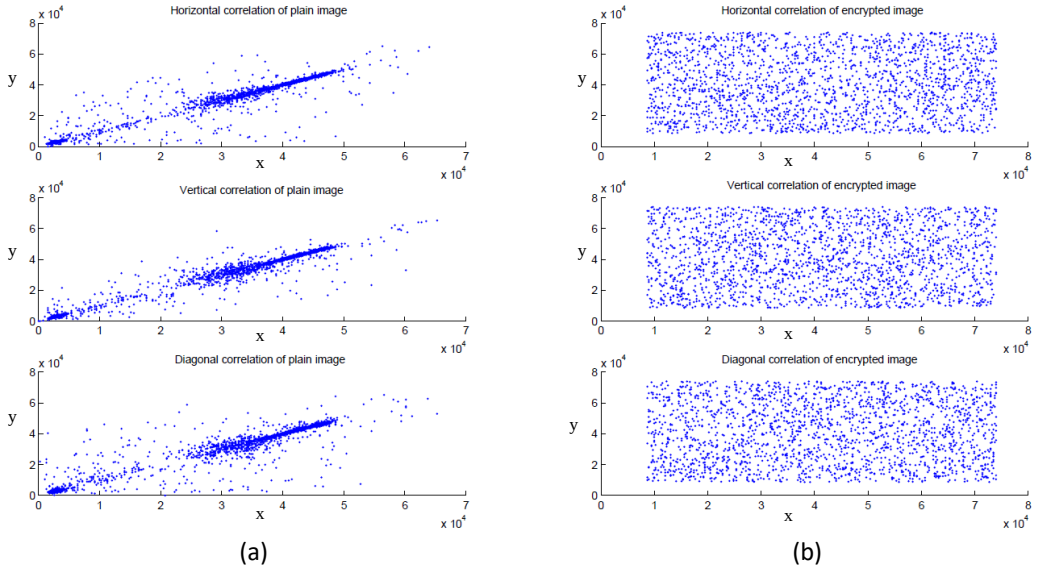



Figure 3.6: Correlation analysis of IGN (a) Plain cameraman image and (b) Encrypted cameraman image

#### 3.4.3.3 Differential analysis

The sensitivity of IGN towards plain image is tested using differential analysis. Table 3.3 shows the average and variance values of NPCR and UACI after 30 independent runs. It is observed that the IGN is extremely sensitive towards small change in the plain image.

Table 3.3: NPCR and UACI analysis of IGN

| Image | NPCR | UACI |
|---|---|---|
| Cameraman | $0.9964 \pm 0.0004$ | $0.3349 \pm 0.0012$ |
| Lena | $0.9963 \pm 0.0004$ | $0.3341 \pm 0.0020$ |
| Baboon | $0.9965 \pm 0.0005$ | $0.3342 \pm 0.0016$ |
| Pirate | $0.9961 \pm 0.0006$ | $0.3344 \pm 0.0022$ |
| Woman | $0.9962 \pm 0.0005$ | $0.3345 \pm 0.0014$ |
| Boat | $0.9964 \pm 0.0005$ | $0.3344 \pm 0.0017$ |
| Airplane | $0.9964 \pm 0.0006$ | $0.3348 \pm 0.0013$ |
| Peppers | $0.9963 \pm 0.0007$ | $0.3336 \pm 0.0023$ |
| House | $0.9964 \pm 0.0006$ | $0.3343 \pm 0.0019$ |
| Lake | $0.9964 \pm 0.0006$ | $0.3341 \pm 0.0019$ |

#### 3.4.3.3.1 Qualitative and Quantitative analysis of NPCR and UACI

Generally, the higher values of NPCR and UACI are considered as a high resistance against differential attack. However, it is not clear how much high values of NPCR and UACI are needed so that it can reflect true randomness of encrypted images [168]. Wu *et al.* [169] designed a mathematical formulation for ideally encrypted images and then derived randomness tests. These tests include expectations, variances, and hypothesis tests for NPCR and UACI to evaluate the performance of encryption techniques against differential attack. To calculate the qualitative score of NPCR under different significance levels, critical NPCR (*i.e.,* $N^*_\rho$) is evaluated as [169]:

$$N^*_\rho = \frac{L_h - \Upsilon^{-1}(\rho)\sqrt{(L_h/W \times H)\Big)}}{L_h + 1}, \tag{3.6}$$

The actual value of NPCR is calculated using Eqs. (1.12) and (1.13). If the actual value NPCR is greater than $N^*_\rho$, then it proves to be an efficient against differential attack in terms of NPCR. Also, the two critical values of UACI (i.e, $U^{*-}_\rho$ and $U^{*+}_\rho$) are used by Wu *et al.* under $\rho$ level of significance, where

$$\begin{aligned} U^{*-}_\rho &= \mu_U - \Upsilon^{-1}(\rho/2)\,\sigma_U, \\ U^{*+}_\rho &= \mu_U - \Upsilon^{-1}(\rho/2)\,\sigma_U, \end{aligned} \tag{3.7}$$

If actual UACI $\in$ [$U^{*-}_\rho$ and $U^{*+}_\rho$], then it is said to be an efficient against differential attacks in terms of UACI. The actual UACI values are computed using Eq. (1.14).

The mean ($\mu_U$) and standard deviation ($\sigma_U$) of UACI are obtained by considering Eqs. (3.8) and (3.9), respectively [170].

$$\mu_U = \frac{L_h + 2}{3L_h + 3}, \tag{3.8}$$

$$\sigma^2_U = \frac{(L_h + 2)\left({L_h}^2 + 2L_h + 3\right)}{18(L_h + 1)^2 L_h \cdot WH}, \tag{3.9}$$

where $L_h$ represents the highest pixel value which is compatible with encrypted image format. $\Upsilon^{-1}$ represents the inverse cumulative density function of standard normal distribution having zero mean and one standard deviation. $W$ and $H$ represent width and height of an image.

NPCR and UACI values of $100$ encrypted images are obtained from IGN by changing one bit of input image reported in Tables 3.4 and 3.5, respectively. The values of NPCR and UACI are significant against the randomness test at a threshold of $5$ % under significance level $\rho = 0.01$ and $\rho = 0.05$.

Table 3.4: Qualitative and quantitative NPCR analysis of IGN

| Image | NPCR score % | | | | |
|---|---|---|---|---|---|
| | Actual NPCR (%) | Theoretical NPCR | | Actual $\sigma$ | Ideal $\sigma$ |
| | | $N^*_{0.01}$ | $N^*_{0.05}$ | | |
| | | 99.5527 | 99.5693 | | 0.0244 |
| Cameraman | 99.64 | 99.6 | 96.0 | 0.0246 | ✓ |
| Lena | 99.63 | 99.2 | 95.4 | 0.0238 | ✓ |
| Baboon | 99.65 | 98.8 | 95.0 | 0.0240 | ✓ |
| Pirate | 99.61 | 98.8 | 95.0 | 0.0254 | ✓ |
| Woman | 99.62 | 99.6 | 95.0 | 0.0242 | ✓ |
| Boat | 99.64 | 99.2 | 97.4 | 0.0243 | ✓ |
| Airplane | 99.64 | 99.3 | 97.6 | 0.0258 | ✓ |
| Peppers | 99.63 | 99.5 | 96.3 | 0.0246 | ✓ |
| House | 99.64 | 99.1 | 96.2 | 0.0239 | ✓ |
| Lake | 99.64 | 99.7 | 95.9 | 0.0240 | ✓ |

The experimental results show that the encrypted image successfully satisfies randomness test at $\rho = 0.01$ and $\rho = 0.05$. The obtained values of NPCR and UACI along with their $\mu$ and $\sigma$ are close towards the ideal values. Therefore, IGN provides significant results against differential attacks.

Table 3.5: Qualitative and Quantitative UACI analysis of IGN

| Image | UACI score % | | | | |
|---|---|---|---|---|---|
| | Actual UACI(%) | Theoretical UACI | | Actual $\sigma$ | Ideal $\sigma$ |
| | | $U^{*-}_{0.01}$/ $U^{*+}_{0.01}$ | $U^{*-}_{0.05}$/ $U^{*+}_{0.05}$ | | |
| | | 33.2255/33.7016 | 33.2824/33.6447 | | 0.0924 |
| Cameraman | 33.49 | 99.9 | 96.0 | 0.0922 | ✓ |
| Lena | 33.41 | 99.8 | 97.2 | 0.0885 | ✓ |
| Baboon | 33.42 | 99.6 | 95.0 | 0.0883 | ✓ |
| Pirate | 33.44 | 99.2 | 95.1 | 0.0968 | ✓ |
| Woman | 33.45 | 99.6 | 95.1 | 0.0964 | ✓ |
| Boat | 33.44 | 99.4 | 97.8 | 0.0897 | ✓ |
| Airplane | 33.48 | 99.3 | 96.8 | 0.0758 | ✓ |
| Peppers | 33.36 | 99.7 | 95.5 | 0.0452 | ✓ |
| House | 33.43 | 99.8 | 95.6 | 0.0452 | ✓ |
| Lake | 33.41 | 99.0 | 95.3 | 0.0452 | ✓ |

#### 3.4.3.4 Confusion and diffusion analysis

An image encryption technique is said to be an efficient if it exhibits the confusion and diffusion properties. The three measures are used to evaluate the performance of confusion and diffusion properties [159]. For a positive integer $n$, $B_{K_1}^{(n)}, B_{K_2}^{(n)}, \ldots, B_{K_n}^{(n)} \in \mathbb{Z}_2^n$. Where $\mathbb{Z}_2^n$ represents $n$ dimensional vector space. For a function $f : Z_2^n \longrightarrow Z_2^m$, denoted by $f_i(1 \leq j \leq m)$, the function $Z_2^n \longrightarrow Z_2^m$ such that

$$f(x,y) = (f(x,y)_m, f(x,y)_{m-1}, \ldots, f(x,y)_2, f(x,y)_1). \tag{3.10}$$

Here, $(x, y)$ represents pixel value of an image.

##### 3.4.3.4.1 Completeness

In image encryption, a function $f : Z_2^n \longrightarrow Z_2^m$ is said to be complete if the value of each encrypted bit depends upon its input bits [159]. It is computed as [171]:

$$d_1 = \sum_{(x,y)\in\mathbb{Z}_2^n} f(x,y) \oplus f((x,y) \oplus B_{K_i}^n). \tag{3.11}$$

The encryption technique is said to be complete if the bit value of $d_1$ is greater than $0$, $\forall i(1 \leq i \leq n)$. Table 3.6 shows the analysis of completeness.

Table 3.6: Completeness, AE, and SAC analysis of IGN

| Technique | Completeness ($d_1$) | AE ($d_2$) | SAC ($d_3$) |
|---|---|---|---|
| IGN | 1 | 0.5672 | 0.9998 |
| GA | 1 | 0.5367 | 0.9984 |
| ACO | 1 | 0.5164 | 0.9989 |
| WDICA | 1 | 0.4981 | 0.9993 |
| GDNA | 1 | 0.5259 | 0.9995 |
| DHS | 1 | 0.5334 | 0.9996 |

##### 3.4.3.4.2 Avalanche effect

Avalanche effect ($AE$) states that for a given function, $Z_2^n \longrightarrow Z_2^m$, to exhibit $AE$, a mean of one half of the output bits should change if a single input bit got changed. $AE$ is computed as [171]:

$$d_2 = \sum_{(x,y)\in\mathbb{Z}_2^n} h_w(f(x,y) \oplus f((x,y) \oplus B_{K_i}^n)). \tag{3.12}$$

Here, $h_w$ represents the hamming weight. An encryption technique achieves $AE$ if the value of $d_2$ is $m2^{n-1}$, $\forall i(1 \leq i \leq n)$. Table 3.6 depicts the $AE$ analysis of IGN.

Additionally, to evaluate $AE$ of IGN, two cases are taken (i.e, (i) changing a bit in secret key, $B_k$) and (ii) changing a bit in input image).

Table 3.7 shows the analysis of $AE$ by considering the change in an input image and $B_k$ by changing a single bit value. It can be observed from the table that the IGN satisfies the $AE$ analysis. Figure 3.7 shows the effect of one bit change in input image.

Table 3.7: Avalanche effect analysis of IGN

| Image | One bit change in input image (%) | One bit change in the key (%) |
|---|---|---|
| Cameraman | 56.72 | 56.86 |
| Lena | 54.53 | 53.38 |
| Baboon | 55.95 | 54.34 |
| Pirate | 56.76 | 56.49 |
| Woman | 54.67 | 55.67 |
| Boat | 55.98 | 56.16 |
| Airplane | 53.81 | 54.41 |
| Peppers | 56.64 | 52.98 |
| House | 54.43 | 53.66 |
| Lake | 53.57 | 54.54 |

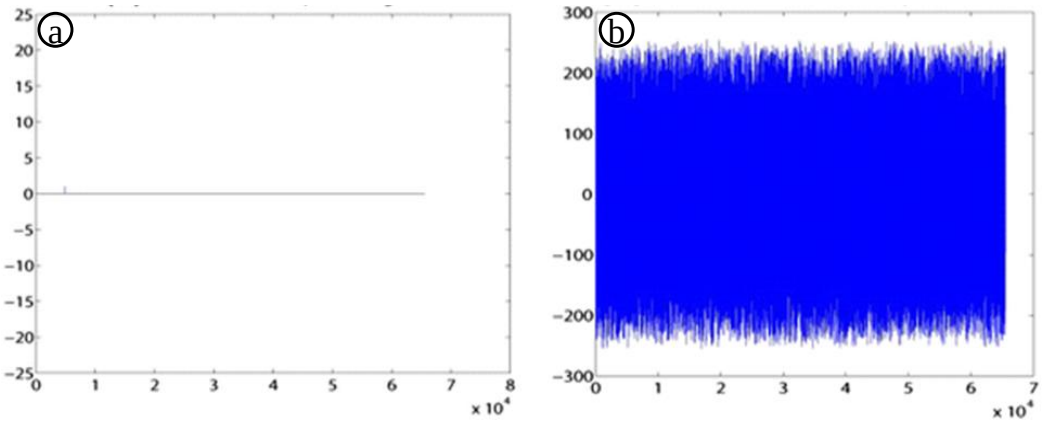



Figure 3.7: Avalanche effect of IGN (a) Difference in input image and (b) Difference in encrypted image

#### 3.4.3.4.3 Strict avalanche criterion analysis

Strict avalanche criterion ($SAC$) considers both completeness and $AE$. It is satisfied when modification in single input bit changes the output bits with half probability. $SAC$ is computed as [171]:

$$d_3 = \sum_{(x,y)\in\mathbb{Z}_2^n} f(x,y) \oplus f((x,y) \oplus B^n_{K_i}). \tag{3.13}$$

The encryption technique satisfies $SAC$ if the value of $d_3 = 2^{n-1}$, $\forall i(1 \leq i \leq n)$. $SAC$ analysis of IGN is shown in Table 3.6. IGN satisfies completeness, $AE$, and $SAC$.

#### 3.4.3.5 Secret key analysis

To resist against brute-force attacks, it is important to evaluate key space and sensitivity of IGN [60].

##### 3.4.3.5.1 Secret key space

The key space of a secret key should be large enough that it cannot be discovered easily. In IGN, the size of secret key $(B_k)$ is same as that of input image $(I)$. The size of $B_k$ is $256 \times 256$ bytes which means the key space of $B_k$ is $256! \times 256! \approx 2^{3369}$ bytes. The key size of $B_k$ is about $(2^{3369})^8 = 2^{26952}$, *i.e.,* about 26952 bits. Thus, IGN has a huge key space to resist against brute-force attacks.

##### 3.4.3.5.2 Secret key sensitivity

The image encryption technique must be sensitive towards the initial values of secret key. Figure 3.8 demonstrates the difference between $E$ and $E'$.

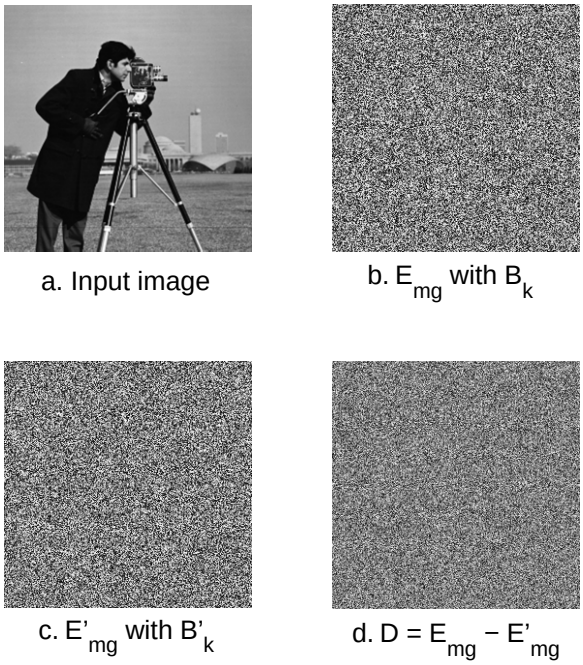



Figure 3.8: Secret key sensitivity of IGN (a) Original cameraman image, (b) Encrypted image with $B_k$, (c) Encrypted image with $B_k{}'$, and (d) Difference between (b) and (c)

To evaluate the sensitivity of secret key, select the input image and generate secret key $(B_k)$. The second secret key ($B_k{}'$) is generated with the difference of one pixel. The two encrypted images are generated by utilizing $B_k$ and $B_k{}'$. Finally, the difference between these encrypted images is computed. $E$ and $E'$ are two encrypted images of same plain image are generated using different secret keys such as $B_k$ and $B_k{}'$ with difference of a single pixel. Table 3.8 shows the difference between $E$ and $E'$ using $B_k$ and $B_k{}'$. It can be observed from table that the IGN is extremely sensitive towards initial conditions.

Table 3.8: Difference between $E$ and $E'$ using $B_k$ and $B_k{}'$

| Gray images | Difference | Color images | Difference |
|---|---|---|---|
| Cameraman | $99.9893$ | Boat | $99.900$ |
| Lena | $99.976$ | Airplane | $99.928$ |
| Baboon | $99.995$ | Peppers | $99.944$ |
| Pirate | $99.938$ | House | $99.996$ |
| Woman | $99.981$ | Lake | $99.974$ |

#### 3.4.3.6 Occlusion attack analysis

As known in prior, encrypted images are prone to various attacks, as images are transfered over internet. Therefore, an image encryption technique is said to be efficient if it is able to resist various attacks. In this thesis, an occlusion attack is also considered to evaluate the effectiveness of IDN technique.

To test the robustness of IGN against data loss, the occlusion attack has been evaluated. It is utilized to assess the capacity of restoring actual images from encrypted images. PSNR is used to evaluate the quality of attacked cipher image.

Figures 3.9 (a)-(d) show the cipher images with four different occlusions, *i.e.,* $1/16$, $1/8$, $1/4$, and $1/2$, respectively. Figures 3.9 (e)-(h) show their corresponding recovered images. It is observed from Figure 3.9 (h) that when half of the data is lost, the recovered image is still recognizable.

Table 3.9 shows the quantitative analysis of occlusion attack on IGN. The values of PSNR for recovered images corresponding to $1/16$, $1/8$, $1/4$, and $1/2$ occlusions are $38.72$, $35.63$, $33.45$, and $28.74$, respectively. The values of NPCR indicate that how much pixels has been changed in recovered image. It can also be concluded that NPCR values of recovered images obtained from IGN are $13.7$ %, $26.9$ %, $41.5$ % and $83.9$ % . Therefore, IGN has an ability to resist against occlusion attacks significantly.

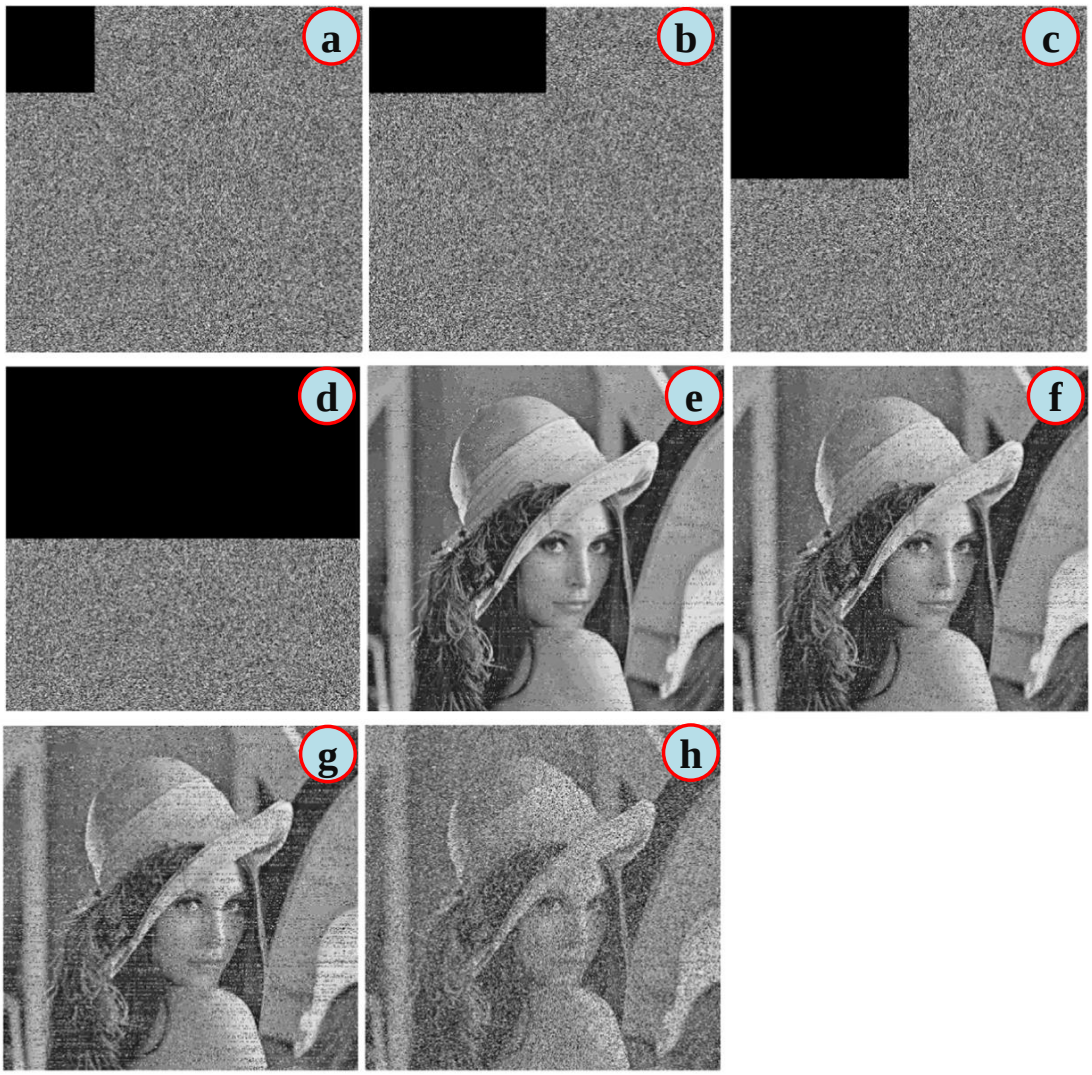



Figure 3.9: Occlusion attack analysis of IGN (a) Cipher image (1/12 occlusion), (b) Cipher image (1/8 occlusion), (c) Cipher image (1/4 occlusion), (d) Cipher image (1/2 occlusion), (e) Recovered image (1/12 occlusion), (f) Recovered image (1/8 occlusion), (g) Recovered image (1/4 occlusion), and (h) Recovered image (1/2 occlusion)

Table 3.9: Occlusion attack analysis of IGN

| Occlusion | NPCR | UACI | PSNR |
|---|---|---|---|
| 1/16 | 13.7 % | 2.4 | 38.72 |
| 1/8 | 26.9 % | 4.2 | 35.63 |
| 1/4 | 41.5 % | 8.5 | 33.45 |
| 1/2 | 83.9 % | 16.8 | 28.74 |

#### 3.4.3.7 Noise attack analysis

IGN is also tested against the noise attacks. As noise present in the encrypted image makes difficult to recover the actual image from encrypted image. Gaussian white noise is added in the encrypted lena image $(E)$.

$$E_n = imnoise(E,' gaussian', m, var), \tag{3.14}$$

where $E_n$ represents the noise affected encrypted image. $m$ and $var$ represent mean and variance, respectively.

Figures 3.10 (a)-(c) show the encrypted images with three different noise levels, *i.e.,* $0.0001$, $0.0003$, and $0.0005$, respectively. Figures 3.10 (d)-(f) show their corresponding decrypted images $(D)$. It has been observed from Figure 3.10 (f) that the decrypted image has worst quality for $0.0005$. The decrypted image can be roughly recognized by observing the outlines or edges. Therefore, IGN has an ability to resist against noise attacks.

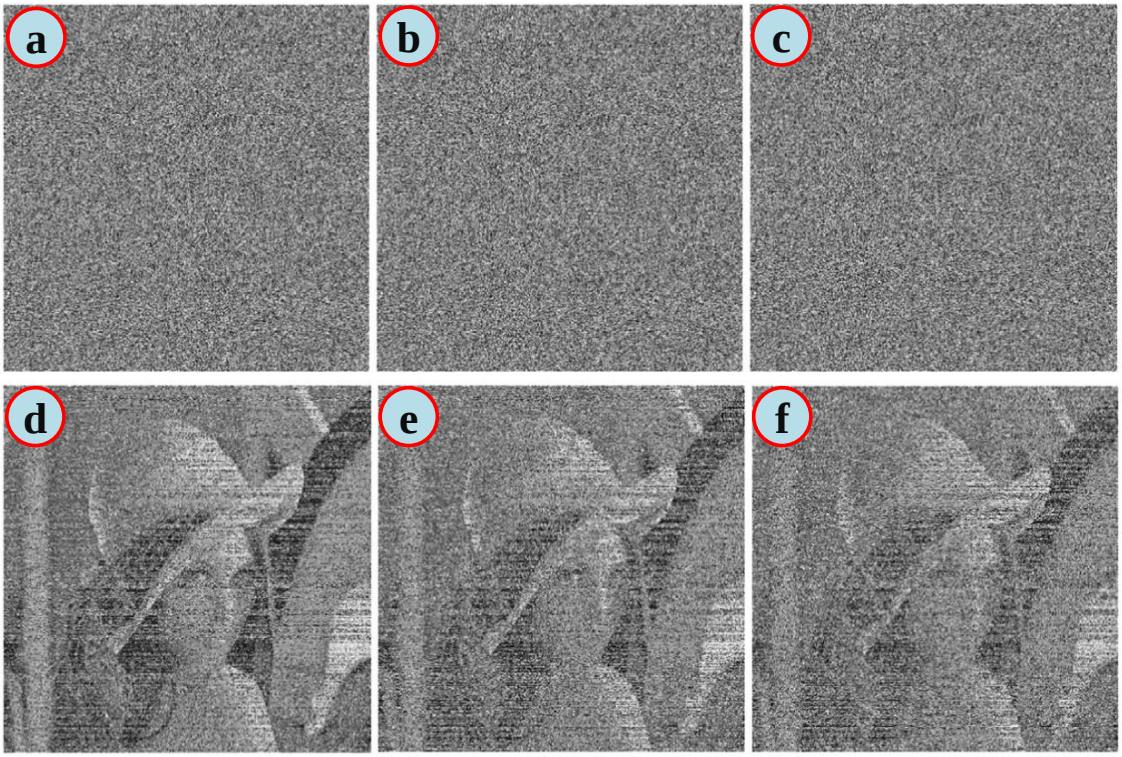



Figure 3.10: Noise attack analysis of IGN (a) $E_n$ with variance=0.0001, (b) $E_n$ with variance=0.0003, (c) $E_n$ with variance=0.0005, (d) $D$ with variance=0.0001, (e) $D$ with variance=0.0003, and (f) $D$ with variance=0.0005

### 3.4.4 Comparative analysis

Tables 3.10 and 3.11 show the comparative analysis between IGN and other techniques in terms of entropy and correlation coefficient for gray images. Tables 3.12 and 3.13 show the comparison of IGN with other techniques on the basis of entropy and correlation coefficient for color images. From Tables 3.10 and 3.12, it can be observed that IGN provides better entropy than other techniques. It means that every pixel of encrypted images carries the same amount of information. Therefore, the attacker cannot get any statistical information about an original image.

Table 3.10: Entropy analysis of IGN for gray images

| Image | GA | ACO | WDICA | GDNA | DHS | IGN |
|---|---|---|---|---|---|---|
| Cameraman | 7.9971 | 7.9484 | 7.9352 | 7.8693 | 7.8739 | 7.9975 |
| Lena | 7.9971 | 7.9540 | 7.9339 | 7.8753 | 7.8702 | 7.9975 |
| Baboon | 7.9903 | 7.9554 | 7.9378 | 7.8688 | 7.8727 | 7.9978 |
| Pirate | 7.9974 | 7.9575 | 7.9359 | 7.8719 | 7.8947 | 7.9976 |
| Woman | 7.9973 | 7.9500 | 7.9361 | 7.8696 | 7.8756 | 7.9976 |

Table 3.11: Correlation analysis of IGN for gray images

| Image | Metric | GA | ACO | WDICA | GDNA | DHS | IGN |
|---|---|---|---|---|---|---|---|
| Cameraman | Hcorr | 0.0060 | 0.0137 | 0.0089 | 0.0152 | 0.0108 | −0.0001 |
| | Dcorr | 0.0037 | 0.0067 | 0.0138 | 0.0098 | 0.0068 | 0.0036 |
| | Vcorr | 0.0113 | 0.0136 | 0.0073 | 0.0120 | 0.0150 | 0.0073 |
| Lena | Hcorr | 0.0018 | 0.0064 | −0.0149 | −0.0126 | −0.0220 | 0.0012 |
| | Dcorr | −0.0047 | 0.0051 | −0.0119 | −0.0082 | −0.0127 | −0.0063 |
| | Vcorr | −0.0074 | −0.0095 | 0.0148 | −0.0052 | −0.0141 | 0.0058 |
| Baboon | Hcorr | 0.0020 | 0.0126 | −0.0137 | −0.0152 | −0.0040 | 0.0001 |
| | Dcorr | −0.0002 | 0.0093 | −0.0013 | −0.0098 | −0.0081 | −0.0008 |
| | Vcorr | 0.0038 | −0.0058 | 0.0035 | −0.0120 | −0.0113 | 0.0002 |
| Pirate | Hcorr | −0.0005 | −0.0016 | −0.0137 | −0.0110 | −0.0107 | 0.0022 |
| | Dcorr | −0.0037 | 0.0058 | −0.0013 | −0.0092 | −0.0052 | 0.0006 |
| | Vcorr | 0.0047 | −0.0150 | 0.0035 | −0.0095 | −0.0160 | −0.0029 |
| Woman | Hcorr | 0.0010 | 0.0053 | −0.0173 | −0.0181 | −0.0087 | 0.0032 |
| | Dcorr | −0.0009 | 0.0064 | −0.0170 | −0.0164 | −0.0211 | 0.0028 |
| | Vcorr | −0.0011 | −0.0137 | 0.0171 | −0.0120 | −0.0145 | 0.0023 |

From Tables 3.11 and 3.13, it can be seen that IGN has minimum horizontal, diagonal, and vertical correlation in most of the cases. It implies that IGN creates better random natured encrypted images. Hence, the attacker cannot easily find the relation among

adjacent pixels to extract the statistical information. Thus, IGN provides better security against statistical attacks.

Table 3.12: Entropy analysis of IGN for color images

| Image | GA | ACO | WDICA | GDNA | DHS | IGN |
|---|---|---|---|---|---|---|
| Boat | 7.9972 | 7.9562 | 7.9357 | 7.8708 | 7.8743 | 7.9977 |
| Airplane | 7.9969 | 7.9499 | 7.9344 | 7.8758 | 7.8712 | 7.9975 |
| Peppers | 7.9970 | 7.9689 | 7.9382 | 7.8901 | 7.8806 | 7.9974 |
| House | 7.9968 | 7.9545 | 7.9364 | 7.8941 | 7.8698 | 7.9977 |
| Lake | 7.9909 | 7.9654 | 7.966 | 7.8802 | 7.8769 | 7.9977 |

Table 3.13: Correlation analysis of IGN for color images

| Image | Metric | GA | ACO | WDICA | GDNA | DHS | IGN |
|---|---|---|---|---|---|---|---|
| Boat | Hcorr | 0.0007 | 0.0058 | 0.0050 | 0.0063 | 0.0077 | 0.0111 |
| | Dcorr | 0.0094 | 0.0019 | 0.0039 | 0.0009 | 0.0034 | 0.0010 |
| | Vcorr | 0.0022 | 0.0085 | 0.0058 | 0.0069 | 0.0090 | 0.0024 |
| Airplane | Hcorr | 0.0062 | 0.0098 | 0.0065 | 0.0015 | 0.0099 | 0.0045 |
| | Dcorr | 0.0042 | 0.0076 | 0.0058 | 0.0027 | 0.0032 | −0.0021 |
| | Vcorr | 0.0006 | 0.0088 | 0.0066 | 0.0066 | 0.0044 | −0.0021 |
| Peppers | Hcorr | 0.0046 | 0.0012 | 0.0020 | 0.0091 | 0.0073 | −0.0015 |
| | Dcorr | 0.0090 | 0.0034 | 0.0053 | 0.0030 | 0.0064 | −0.0046 |
| | Vcorr | 0.0087 | 0.0018 | 0.0037 | 0.0099 | 0.0100 | −0.0041 |
| House | Hcorr | 0.0020 | 0.0055 | 0.0058 | 0.0046 | 0.0018 | −0.0036 |
| | Dcorr | 0.0062 | 0.0016 | 0.0079 | 0.0091 | 0.0007 | −0.0002 |
| | Vcorr | 0.0036 | 0.0011 | 0.0031 | 0.0099 | 0.0074 | 0.0044 |
| Lake | Hcorr | 0.0065 | 0.0053 | 0.0079 | 0.0019 | 0.0003 | −0.0003 |
| | Dcorr | 0.0086 | 0.0056 | 0.0011 | 0.0044 | 0.0068 | −0.0008 |
| | Vcorr | 0.0028 | 0.0095 | 0.0070 | 0.0051 | 0.0011 | −0.0076 |

As differential attack is a kind of chosen plaintext attack that states that an attacker adaptively selects some plain images and evaluate the corresponding encrypted images by accessing the encryption technique. Thereafter, evaluate the difference between actual encrypted image and the encrypted image obtained by an attacker. The attacker can only select the plain images which are different from the actual image of the encrypted he tries to crack. Therefore, differential attack is used to evaluate the affect of one pixel value changed in an input image on the encrypted image, rather than the input image on the decrypted image.

Tables 3.14 and 3.16 show the performance comparison of IGN with the existing techniques in terms of NPCR and UACI for gray and color images, respectively. From tables,

it can seen that IGN provides better NPCR and UACI as compare to other technique. It means that IGN generates a totally different image if the slight change is made in the plain image. Therefore, IGN is able to resist differential attack in an efficient manner.

Table 3.14: Comparative analysis of IGN using NPCR and UACI for gray images

| Image | Metric | GA | ACO | WDICA | GDNA | DHS | IGN |
|---|---|---|---|---|---|---|---|
| Cameraman | NPCR | 0.9959 | 0.9731 | 0.9960 | 0.9961 | 0.9946 | 0.9964 |
| | UACI | 0.3339 | 0.3345 | 0.3341 | 0.3346 | 0.3345 | 0.3349 |
| Lena | NPCR | 0.9969 | 0.9732 | 0.9962 | 0.9954 | 0.9956 | 0.9963 |
| | UACI | 0.3314 | 0.3341 | 0.3344 | 0.3347 | 0.3349 | 0.3341 |
| Baboon | NPCR | 0.9962 | 0.9730 | 0.9960 | 0.9959 | 0.9943 | 0.9965 |
| | UACI | 0.3318 | 0.3347 | 0.3352 | 0.3346 | 0.3351 | 0.3342 |
| Pirate | NPCR | 0.9961 | 0.9734 | 0.9960 | 0.9963 | 0.9959 | 0.9961 |
| | UACI | 0.3316 | 0.3350 | 0.3347 | 0.3351 | 0.3346 | 0.3344 |
| Woman | NPCR | 0.9952 | 0.9733 | 0.9959 | 0.9964 | 0.9960 | 0.9962 |
| | UACI | 0.3318 | 0.3327 | 0.3352 | 0.3339 | 0.3352 | 0.3345 |

Table 3.15: Comparative analysis of IGN using PSNR and MAE for gray images

| Image | Metric | GA | ACO | WDICA | GDNA | DHS | IGN |
|---|---|---|---|---|---|---|---|
| Cameraman | PSNR | 66.6628 | 32.2670 | 64.9113 | 64.6482 | 64.9151 | 87.0372 |
| | MAE | 79.2275 | 81.6399 | 79.8161 | 80.9763 | 80.9788 | 129.768 |
| Lena | PSNR | 65.2259 | 32.2669 | 64.3108 | 64.1811 | 64.6722 | 92.8158 |
| | MAE | 72.8957 | 75.0392 | 73.8975 | 74.4053 | 74.3374 | 78.5998 |
| Baboon | PSNR | 65.3524 | 32.7895 | 64.2486 | 64.1947 | 64.6795 | 88.6406 |
| | MAE | 69.5667 | 71.9020 | 70.1388 | 70.8065 | 71.0397 | 130.120 |
| Pirate | PSNR | 65.1093 | 32.9856 | 64.3411 | 64.2271 | 64.6728 | 94.1561 |
| | MAE | 73.4669 | 75.9907 | 75.2574 | 75.5613 | 75.6586 | 80.9300 |
| Woman | PSNR | 64.6272 | 32.5670 | 64.4623 | 64.0407 | 64.4623 | 85.0297 |
| | MAE | 80.1179 | 82.6779 | 82.0500 | 82.7718 | 82.6055 | 86.6293 |

Tables 3.15 and 3.17 show the performance comparison of IGN with other techniques in terms of PSNR and MAE for gray and color images, respectively. From tables, it can be observed that IGN also provides better PSNR than other techniques. It shows that IGN produces better quality decrypted images. IGN also provides significant MAE as compare to other techniques. It implies that IGN generates totally different encrypted images from the input images. So, the attacker cannot find any similarity between input and encrypted images.

Table 3.16: Comparative analysis of IGN using NPCR and UACI for color images

| Image | Metric | GA | ACO | WDICA | GDNA | DHS | IGN |
|---|---|---|---|---|---|---|---|
| Boat | NPCR | 0.9953 | 0.9733 | 0.9961 | 0.9951 | 0.9948 | 0.9964 |
| | UACI | 0.3329 | 0.3242 | 0.3343 | 0.3336 | 0.3341 | 0.3344 |
| Airplane | NPCR | 0.9959 | 0.9701 | 0.9960 | 0.9957 | 0.9959 | 0.9964 |
| | UACI | 0.3324 | 0.3338 | 0.3345 | 0.3340 | 0.3349 | 0.3348 |
| Peppers | NPCR | 0.9962 | 0.9799 | 0.9960 | 0.9948 | 0.9949 | 0.9963 |
| | UACI | 0.3328 | 0.3340 | 0.3349 | 0.3344 | 0.3350 | 0.3336 |
| House | NPCR | 0.9963 | 0.9799 | 0.9960 | 0.9963 | 0.9960 | 0.9964 |
| | UACI | 0.3335 | 0.3340 | 0.3337 | 0.3349 | 0.3343 | 0.3343 |
| Lake | NPCR | 0.9962 | 0.9744 | 0.9963 | 0.9964 | 0.9960 | 0.9964 |
| | UACI | 0.3342 | 0.3329 | 0.3325 | 0.3340 | 0.3350 | 0.3341 |

Table 3.17: Comparative analysis of IGN using PSNR and MAE for color images

| Image | Metric | GA | ACO | WDICA | GDNA | DHS | IGN |
|---|---|---|---|---|---|---|---|
| Boat | PSNR | 66.6582 | 32.6270 | 64.9356 | 64.9817 | 64.5890 | 82.6506 |
| | MAE | 79.7522 | 79.6939 | 80.8161 | 81.9663 | 80.9788 | 88.9704 |
| Airplane | PSNR | 65.1039 | 32.3801 | 64.0089 | 64.5630 | 64.3894 | 88.5357 |
| | MAE | 76.8407 | 73.1582 | 79.9982 | 78.4503 | 82.1374 | 85.2984 |
| Peppers | PSNR | 65.6401 | 32.2407 | 64.7001 | 64.1088 | 64.8990 | 89.3253 |
| | MAE | 69.4003 | 73.1089 | 73.1338 | 79.4815 | 74.2897 | 85.4931 |
| House | PSNR | 65.8754 | 32.2850 | 64.0017 | 80.5674 | 64.1856 | 64.9809 | 89.2722 |
| | MAE | 72.0900 | 78.7088 | | 77.7113 | 81.3586 | 88.9809 |
| Lake | PSNR | 64.9921 | 32.3049 | 64.4587 | 64.1197 | 64.9043 | 84.9844 |
| | MAE | 85.9025 | 86.2790 | 80.1520 | 84.6210 | 82.7846 | 147.8469 |

Execution time ($ET$) is measured as the time (in seconds) taken to execute a given image encryption technique. It is the aggregation of Compile ($CT$) and Runtime time ($RT$) of the given technique. The 'tic' and 'toc' operators in MATLAB script have been used to evaluate $ET$. Tables 3.18 and 3.19 show the analysis of $ET$ in seconds for encryption and decryption process, respectively. IGN takes lesser execution time as compared to others. The mean reduction in $ET$ by using IGN over the existing techniques is approximately $1.3487$. Also, due to iterative process of meta-heuristic based image encryption techniques, these techniques take more time whenever image size becomes larger (see Table 3.18). However, image size does not effect the execution time of decryption process. It is observed from Tables 3.18 and 3.19 that IGN is computationally faster than the existing image encryption techniques. $\pm$ represents the variation in $ET$. Therefore, $4.1 \pm 3.1$ indicates that the minimum and maximum execution times are found to be $1.0$ and $7.2$, respectively.

Table 3.18: Execution time analysis of IGN in terms of encryption process

| Image | Size | GA | ACO | WDICA | GDNA | DHS | IGN |
|---|---|---|---|---|---|---|---|
| Cameraman | $256 \times 256$ | 11.92 | 14.32 | 9.73 | 8.53 | 7.62 | 7.12 |
| Lena | $512 \times 512$ | 17.86 | 23.95 | 15.94 | 14.63 | 17.53 | 12.32 |
| Baboon | $1024 \times 1024$ | 37.57 | 28.66 | 26.75 | 29.85 | 25.95 | 21.53 |
| Pirate | $2048 \times 1040$ | 58.71 | 47.71 | 49.37 | 51.26 | 46.25 | 41.63 |
| Woman | $2048 \times 2048$ | 91.41 | 78.11 | 81.01 | 78.31 | 82.49 | 73.55 |
| Boat | $256 \times 256$ | 17.94 | 13.24 | 11.43 | 9.73 | 8.83 | 7.62 |
| Airplane | $512 \times 512$ | 28.87 | 23.95 | 24.55 | 19.64 | 23.54 | 17.43 |
| Peppers | $1024 \times 1024$ | 38.29 | 29.78 | 27.27 | 32.37 | 26.95 | 21.82 |
| House | $2048 \times 1040$ | 59.11 | 48.31 | 46.39 | 53.77 | 56.16 | 46.33 |
| Lake | $2048 \times 2048$ | 94.61 | 81.01 | 85.98 | 87.89 | 78.38 | 72.76 |

Table 3.19: Execution time analysis of IGN in terms of decryption process

| Image | Size | GA | ACO | WDICA | GDNA | DHS | IGN |
|---|---|---|---|---|---|---|---|
| Cameraman | $256 \times 256$ | 0.0471 | 0.0381 | 0.0294 | 0.0320 | 0.0271 | 0.0181 |
| Lena | $512 \times 512$ | 0.0413 | 0.0372 | 0.0292 | 0.0393 | 0.0292 | 0.0191 |
| Baboon | $1024 \times 1024$ | 0.0432 | 0.0352 | 0.0292 | 0.0383 | 0.0252 | 0.0211 |
| Pirate | $2048 \times 1040$ | 0.0461 | 0.0342 | 0.0242 | 0.0363 | 0.0302 | 0.0221 |
| Woman | $2048 \times 2048$ | 0.0352 | 0.0352 | 0.0312 | 0.0382 | 0.0262 | 0.0211 |
| Boat | $256 \times 256$ | 0.0393 | 0.0352 | 0.0302 | 0.0363 | 0.0312 | 0.0211 |
| Airplane | $512 \times 512$ | 0.0372 | 0.0352 | 0.0212 | 0.0413 | 0.0291 | 0.0201 |
| Peppers | $1024 \times 1024$ | 0.0432 | 0.0372 | 0.0292 | 0.0393 | 0.0282 | 0.0211 |
| House | $2048 \times 1040$ | 0.0412 | 0.0322 | 0.0282 | 0.0413 | 0.0292 | 0.0191 |
| Lake | $2048 \times 2048$ | 0.04131 | 0.03925 | 0.03121 | 0.0382 | 0.0321 | 0.0181 |

## 3.5 Summary

In this chapter, IGN is proposed which has an ability to tune the required initial parameters of beta chaotic map for a secure key generation. The tuning of parameters has been done through GA using multi-objective fitness function. Moreover, the encryption has been carried out on sub-bands of an image instead of a plain image, which further enhances the security. IGN is tested on ten well-known benchmark images. It also provides significant quality of decrypted images. To test the security of IGN, the various experiments have been carried out such as statistical attack, differential attack, secret key, noise attack, and occlusion attack analyses. The experimental results have shown that IGN has better performance as compared to the existing image encryption techniques.

# Chapter 4

# Differential evolution based image encryption

## 4.1 Introduction

The main contribution of this chapter is to propose an efficient Image encryption technique based on Differential evolution, namely IDN. It uses differential evolution to tune the initial parameters of beta chaotic map . IDN also uses nonsubsampled contourlet transform and Arnold transform. Arnold transform performs permutation operation in IDN.

## 4.2 Arnold transform

Arnold transform (AT) is a permutation technique which changes the pixel positions of an image without manipulating the pixel's values [172]. In other words, the original energy of an image remains same. It can be mathematically defined as [172]:

$$\begin{bmatrix} \chi' \\ \varphi' \end{bmatrix} = AT\{(\chi, \varphi), S_z(I)\} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} \chi \\ \varphi \end{bmatrix} (mod \ \ S_z(I)) \tag{4.1}$$

Here, $(\chi, \varphi)$ represent coordinates of input image. $(\chi', \varphi')$ show the changed positions of pixel-coordinates of an image. $S_z(I)$ is the size of an image. $\begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}$ is the 2-D scrambling matrix.

After transformation, an original image is changed into another image which is noisy in appearance [173]. Arnold transform is cyclic in nature which means the original image can be recovered after some period of time. Therefore, the number of iterations depend upon the size of an input image. As the size of a given image grows, it will take more

time to recover the original image [174]. Inverse Arnold transform $(AT^{-1})$ is computed to recover an original image. It can be mathematically defined as [172]:

$$\begin{bmatrix}\chi\\ \varphi\end{bmatrix} = AT^{-1}\{(\chi',\varphi'), S_z(I)\} = \begin{bmatrix}2 & -1\\ -1 & 1\end{bmatrix}\begin{bmatrix}\chi'\\ \varphi'\end{bmatrix}(mod\ \ S_z(I)) \tag{4.2}$$

## 4.3 Differential evolution based image encryption technique

An Image encryption technique based on differential evolution (IDN) consists of two main processes. These are encryption and decryption processes.

### 4.3.1 Encryption process

The flow of encryption process is depicted in Figure 4.1. $G$ and $f$ represent generation and fitness function, respectively. Initially, Arnold transform is utilized to permute the pixels position of an input image. Thereafter, the permuted image is divided into sub-bands using NSCT. A pseudo-random key is generated by differential evolution based beta chaotic map to encrypt the sub-bands. Thereafter, inverse NSCT is applied on encrypted sub-bands to obtain the final encrypted image.

**Step 1:** The input image ($I$) having rows ($w$) and columns ($h$).

**Step 2:** Apply Arnold transform on $I$ to generate scrambled image ($I'$). This process is repeated for specified number of iterations ($it$). It is calculated as:

$$I'(\chi',\varphi')^{it} = \begin{bmatrix}1 & 1\\ 1 & 2\end{bmatrix} I(\chi,\varphi)^{it-1}(mod\ \ S_z(I)) \tag{4.3}$$

Here, $it = 1, 2, 3, \ldots, n$ represents the number of iterations.

**Step 3:** Apply NSCT on scrambled image ($I'$) with first level of decomposition (*i.e.*, $dl = 1$) which decomposes $I'$ into three sub-bands. These sub-bands are High-frequency sub-band $1$ ($H^1_{w,h}$), High-frequency sub-band $2$ ($H^2_{w,h}$), and Low-frequency sub-band ($H^0_{w,h}$). The size of each sub-band is same as of $I$. The sub-bands of $I'$ are obtained through NSCT, which are computed as:

$$[H^0_{w,h}, H^1_{w,h}, H^2_{w,h}] \leftarrow NSCT(I', dl) \tag{4.4}$$

**Step 4:** Beta chaotic map generates a secret key $B_k$ using Eqs. (1.3)-(1.7). However, these equations require efficient tuning of parameters. Therefore, differential evolution is used to optimize these parameters. The size of $B_k$ is same as the size of sub-bands.

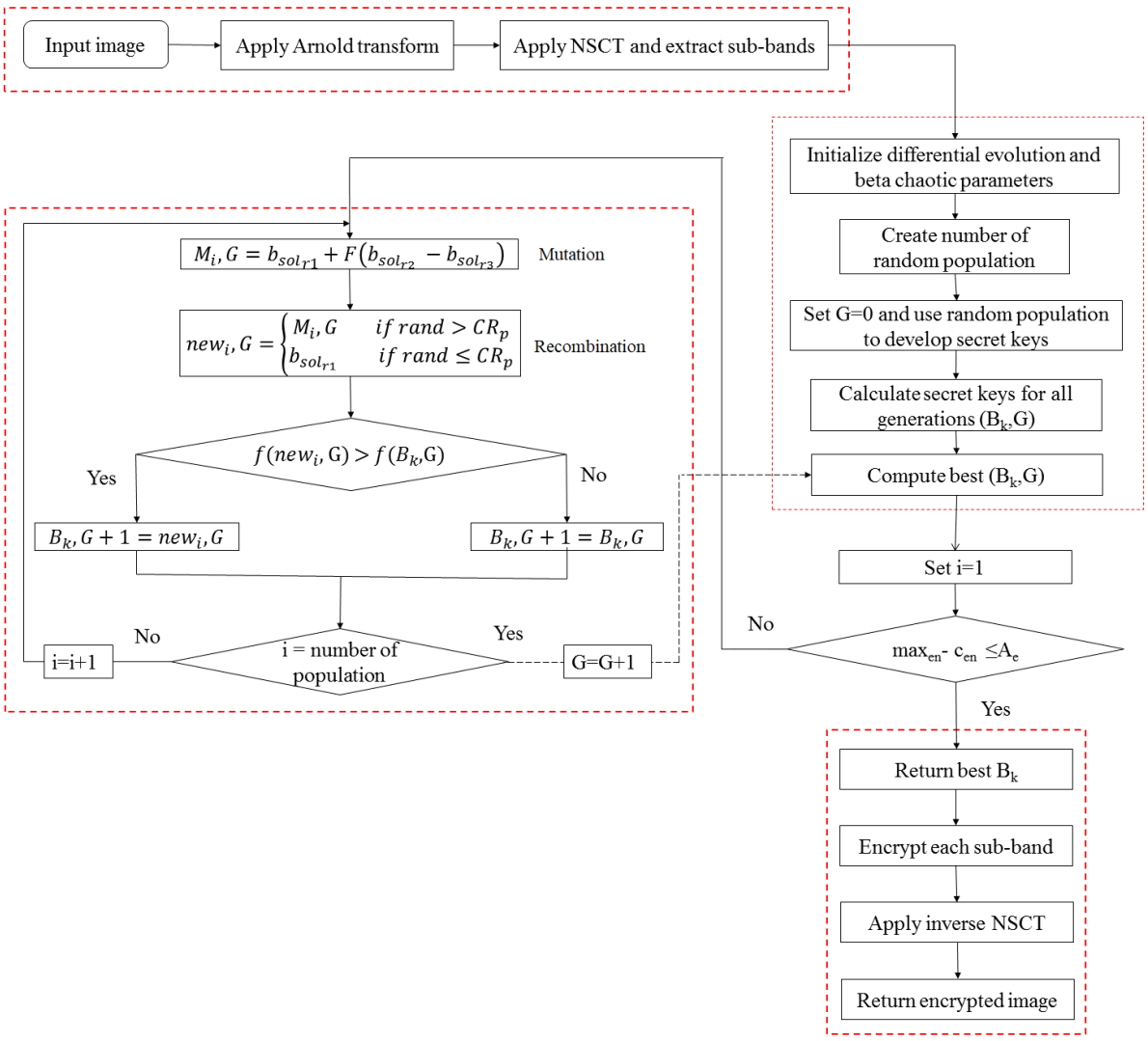



Figure 4.1: Flowchart of differential evolution based image encryption

**Step 4.1 Population initialization:** In this step, differential evolution procedure is initiated by developing its initial population randomly using normal distribution with mean ($\mu$=0) and variance ($\sigma^2 = 1$). Each random solution ($c_s$) have size 10. The lower bound and upper bound values lie between 0 and 1, respectively. First, nine values of each $c_s$ are assigned to the parameters of beta chaotic map to generate a key and last value is used as encryption factor ($\alpha_e$).

**Step 4.2 Fitness function:** Entropy [59] is used as a fitness function by differential evolution to evaluate the best solution. The main objective of fitness function is to obtain a solution which has the maximum entropy of an encrypted image. The fitness function ($f$) is computed using Eq. (1.19).

**Step 4.3 Selection operator:** The fitness of $n_1$ encrypted images are calculated and compared with each other. The variable $b_{en}$ stores fitness value of first encrypted image or best fitness value among previously compared two encrypted images. The variable $c_{en}$ stores the fitness value of current encrypted image. Every random solution $c_s(i)$ corresponding to the best entropy store into a best solution ($b_{sol}(i)$) matrix. Therefore, a selection technique is used as [49]:

$$b_{en}(i+1) = \begin{cases} c_{en}(i+1), & \text{if} \quad b_{en}(i) \leq c_{en}(i+1) \\ b_{en}(i), & \text{otherwise} \end{cases} \tag{4.5}$$

**Step 4.4 Mutation and Recombination operator:** Differential evolution develops trial solutions in an efficient way using mutation and recombination operators. Firstly, a weighted difference between two solutions is merged into a third solution to generate a mutant solution. A target solution ($b_{soli}$) is also selected. $b_{soli}$ represents the best solution obtained so far. For each target solution, a mutant solution ($M_i$) is generated as:

$$M_i = b_{sol_{r_1}} + F(b_{sol_{r_2}} - b_{sol_{r_3}}), r_1 \neq r_2 \neq r_3 \neq i \tag{4.6}$$

with randomly chosen indexes $r_1$, $r_2$, $r_3 \in [1, n_1]$. The randomly chosen indexes should be different from target index $i$. Therefore, the minimum value of $n_1$ should be four. $F$ demonstrates the scaling factor (*i.e.,* $F \in [0, 2]$) that controls the amplification of difference between $b_{sol_{r_2}}$ and $b_{sol_{r_3}}$.

Secondly, $b_{soli}$ is mixed with the mutated solution ($M_i$) to yield the trial solution ($new(j)$) using the following scheme:

$$new(j) = \begin{cases} M_i, & if \quad j \neq j0 \quad || \quad rand > CR_p \\ b_{sol_i}, & if \quad j == j0 \quad || \quad rand <= CR_p \end{cases} \tag{4.7}$$

Here, $CR_p$ represents the crossover constant.

The fitness of obtained trial solution is also computed. In case, the generated solution has significant fitness value then the best solution will be replaced with this solution.

**Step 4.5 Termination criteria:** It is not possible to achieve the maximum fitness value ($max_{en}$ *i.e.,* $8$). Therefore, in this work, we have taken Acceptance error ($A_e$) to stop the differential evolution algorithm. The difference between $max_{en}$ and best fitness is compared with $A_e$. If difference is less than $A_e$, differential evolution will repeat its Steps from $4.2$ to $4.4$. Otherwise, differential evolution will automatically return the optimized parameters for encryption and decryption. Termination criteria ($T_c$) can be mathematically computed as:

$$T_c = \begin{cases} 1, & \text{if} \quad max_{en} - c_{en} \leq A_e \\ 0, & \text{otherwise} \end{cases} \tag{4.8}$$

**Step 5:** The secret key $B_k = \{b_{k_1}, b_{k_2}, b_{k_3}, \ldots, b_{k_{r\times c}}\}$ obtained from differential evolution and encryption factor $\alpha_e$ are utilized to encrypt $H^0_{w,h}$, $H^1_{w,h}$, $H^2_{w,h}$ sub-bands as follows:

$$EH^i_{w,h} \leftarrow \mod(\alpha_e \times H^i_{w,h} + (1-\alpha_e) \times B_k, p_k) \tag{4.9}$$

Here, $EH^i_{w,h}$ represent the encrypted sub-bands with $i = 0, 1,$ or $2$. $p_k$ represents the peak pixel value of input image.

**Step 6:** The encrypted image ($E$) is obtained by applying the inverse of NSCT (iNSCT) on encrypted sub-bands (*i.e.*, $EH^0_{w,h}$, $EH^1_{w,h}$, and $EH^2_{w,h}$).

$$E \leftarrow iNSCT(EH^0_{w,h}, EH^1_{w,h}, EH^2_{w,h}) \tag{4.10}$$

### 4.3.2 Decryption process

In this process, secret key and encryption factor ($\alpha_e$) are required to decrypt the image. Therefore, the same optimized parameters which were used to generate secret key in encryption process send to receiver with same encryption factor.

**Step 1:** The encrypted image ($E$) is decomposed using NSCT.

$$[EH^0_{w,h}, EH^1_{w,h}, EH^2_{w,h}] \leftarrow NSCT(E, dl) \tag{4.11}$$

Here, $EH^0_{w,h}$, $EH^1_{w,h}$, and $EH^2_{w,h}$ are encrypted sub-bands.

**Step 2:** The secret key $B_k$ is developed by applying optimistic parameters provided by differential evolution to beta chaotic map using Eqs. (1.3)-(1.7).

**Step 3:** Apply $B_k$ and $\alpha_e$ on each encrypted sub-band to obtain decrypted sub-bands as follows:

$$DH^i_{w,h} \leftarrow (EH^i_{w,h} - (1-\alpha_e) \times B_k)/\alpha_e \tag{4.12}$$

Here, $DH^i_{w,h}$ represent decrypted sub-bands with $i = \{0, 1, 2\}$. $DH^1_{w,h}$ and $DH^2_{w,h}$ represent decrypted high-frequency sub-bands, and $DH^0_{w,h}$ represent decrypted low-frequency sub-band.

**Step 4:** Inverse of NSCT (iNSCT) is applied on the decrypted sub-bands to obtain an intermediate decrypted image ($D'$) as follows:

$$D' = iNSCT(DH^1_{w,h}, DH^2_{w,h}, DH^0_{w,h}) \tag{4.13}$$

This decrypted image ($D'$) is still in the scrambled form. The original image can be recovered by applying the inverse of Arnold transform on $D'$ with the same number of iterations (*i.e.,* $it$).

**Step 5:** Inverse of Arnold transform can be computed as follows:

$$D(\chi', \varphi')^{it} = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix} D'(\chi, \varphi)^{it-1} (mod \ \ S_z(D')) \tag{4.14}$$

where $D$ represents the final decrypted image.

## 4.4 Experimental results and discussion

To validate IDN, five well-known benchmark color images having size of $256 \times 256$ are taken from USC-SIPI dataset [167]. To improve the parameters tuning for beta chaotic map, population size is set to be $30$ for differential evolution as reported in [175]. The experiments have been done on different values of parameters. While doing the experiments, we have found that each time the value of entropy varies. But, after applying the same technique $30$ times, we have found that the entropy values start repeating itself or have minimum variance with the existing evaluated entropies such as $7.6889$ and $7.6890$. Thereafter, the best values of parameters are reported in Table 4.1.

Table 4.1: Parameter setting used for differential evolution based image encryption technique

| Symbol | Meaning | Value |
|---|---|---|
| $dl$ | Decomposition level | 1 |
| $c_s$ | Random solution size | 10 |
| $CR_p$ | Cross-over constant | 0.2 |
| $A_e$ | Acceptance error | $2^{-10}$ |
| $it$ | Number of iterations | 150 |
| $p_k$ | Peak pixel value | 256 |
| $F$ | Scaling Factor | 0.5 |

Figure 4.2 shows the performance evaluation of IDN on color images. It is observed that the input and decrypted images are identical to each other. The histogram of input and encrypted images show that IDN scramble image in such a way that pixels are evenly distributed. From encrypted images, it is observed that IDN does not suffer from silhouette problem. Therefore, IDN provides significant results in terms of visual quality.

### 4.4.1 Performance evolution

The performance of the IDN is evaluated using some well-known parameters such as entropy, peak signal to noise ratio, and mean absolute error.

#### 4.4.1.1 Entropy

To find best entropy, the IDN is applied 30 times on each test image by considering different number of iterations. Table 4.2 shows the best entropy values of various test images with respect to 10 to 150 iterations. It can be observed that the entropy values of all images approach to an ideal value *i.e.* 8. Hence, the pixels are uniformly distributed in an encrypted image and cannot be anticipated easily.

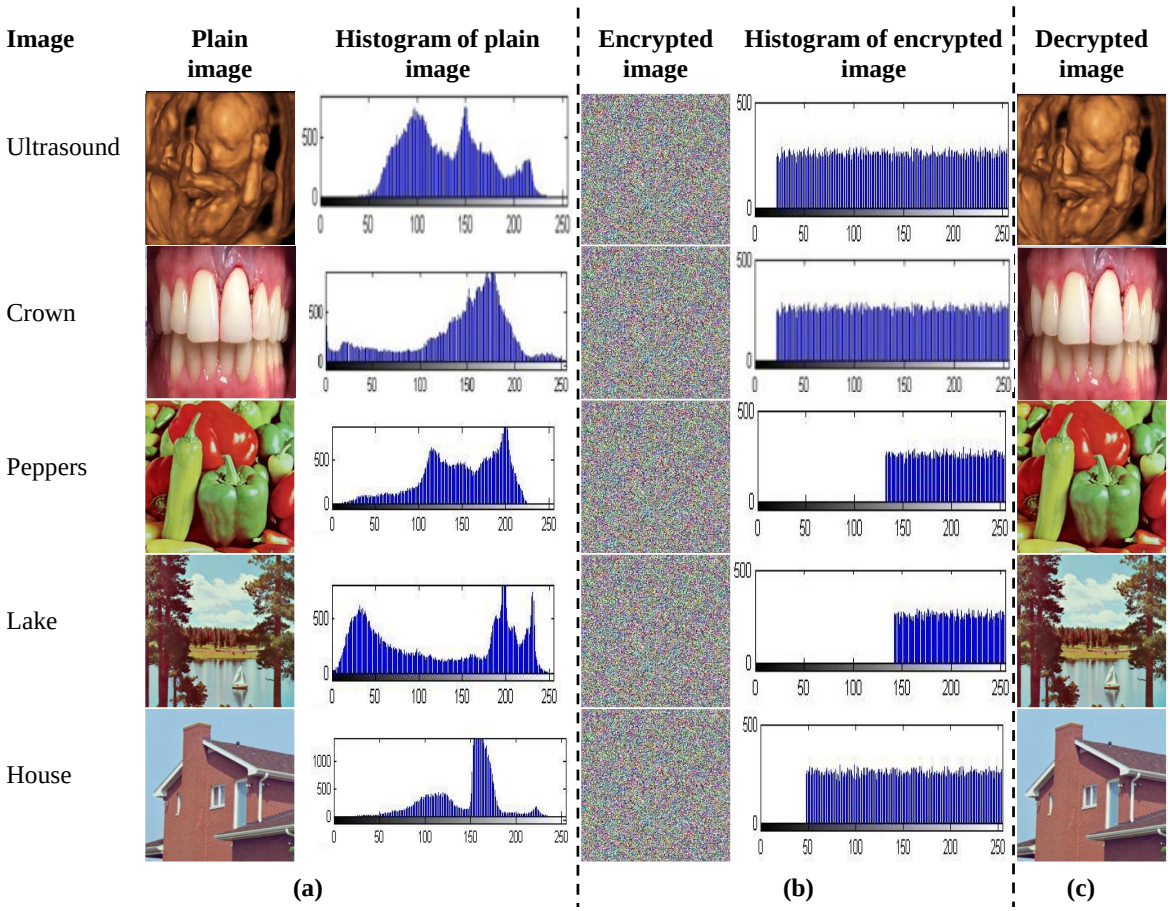



Figure 4.2: Performance evaluation of IDN (a) Original images and corresponding histograms, (b) Encrypted images and corresponding histograms, and (c) Decrypted images

Table 4.2: Best entropies obtained from IDN at different number of iterations

| **Image** | **Iterations** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 10 | 30 | 50 | 70 | 90 | 110 | 130 | 150 |
| Ultrasound | 7.9970 | 7.9971 | 7.9974 | 7.9974 | 7.9976 | 7.9978 | 7.9980 | 7.9980 |
| Crown | 7.9972 | 7.9975 | 7.9978 | 7.9978 | 7.9980 | 7.9980 | 7.9982 | 7.9982 |
| Peppers | 7.9972 | 7.9972 | 7.9974 | 7.9974 | 7.9979 | 7.9979 | 7.9979 | 7.9979 |
| Lake | 7.9974 | 7.9974 | 7.9974 | 7.9974 | 7.9974 | 7.9976 | 7.9976 | 7.9976 |
| House | 7.9975 | 7.9975 | 7.9975 | 7.9975 | 7.9975 | 7.9978 | 7.9978 | 7.9980 |

#### 4.4.1.2 Peak signal to noise ratio

Table 4.3 shows the PSNR of encrypted and decrypted images w.r.t original images. PSNR has minimum values in the first case (*i.e.,* between $I$ and $E$) which means there is a significant difference between encrypted and original images. In the second case (*i.e.,* between $I$ and $D$), the values of PSNR are maximum which means decrypted image is very close to the original image.

#### 4.4.1.3 Mean absolute error

Table 4.3 shows MAE values between plain images and their respective encrypted images. It can be seen from table that the encrypted images are completely different from input images. Therefore, the attacker cannot find any kind of information about an original image from their encrypted image.

Table 4.3: PSNR and MAE analyses of IDN

| **Image** | **PSNR** | | **MAE** |
|---|---|---|---|
| | between $I$ and $E$ | between $I$ and $D$ | between $I$ and $E$ |
| Ultrasound | $-7.1501$ | $80.3210$ | $86.1587$ |
| Crown | $-7.4210$ | $87.4729$ | $83.1778$ |
| Peppers | $-7.1050$ | $89.3253$ | $85.4931$ |
| Lake | $-7.2503$ | $89.2722$ | $88.9809$ |
| House | $-8.9891$ | $84.9844$ | $92.8469$ |

### 4.4.2 Security analysis

To evaluate the effectiveness of IDN against security attacks, various security analyses are considered as follows.

#### 4.4.2.1 Histogram analysis

Figure 4.2 (a) shows the input images with their respective histograms. Figure 4.2 (b) shows encrypted images and their respective histograms. It can be observed from the histograms of encrypted images that the pixels are uniformly distributed. Therefore, it is hard to find any information from these encrypted images.

#### 4.4.2.2 Correlation coefficient

To analyze the correlation analysis of IDN, three correlations namely horizontal, vertical, and diagonal have been evaluated using Eqs. (1.15)-(1.18). Table 4.4 depicts the horizontal, vertical, and diagonal correlation of test images and their ciphered images. It is

observed that attacker can hardly make any kind of relationship between pixels to break the algorithm.

Table 4.4: Correlation analysis of IDN

| Image | Plain image | | | Cipherd image | | |
|---|---|---|---|---|---|---|
| | Horizontal | Vertical | Diagonal | Horizontal | Vertical | Diagonal |
| Ultrasound | 0.9012 | 0.9820 | 0.9314 | −0.0003 | 0.0023 | 0.0065 |
| Crown | 0.9216 | 0.9207 | 0.9156 | 0.0015 | −0.0048 | 0.0042 |
| Peppers | 0.8701 | 0.8411 | 0.7889 | 0.0001 | −0.0008 | 0.0002 |
| Lake | 0.9434 | 0.9564 | 0.9134 | 0.0022 | 0.0006 | −0.0029 |
| House | 0.9914 | 0.9925 | 0.9859 | 0.0032 | 0.0028 | 0.0023 |

Figures 4.3 (a), (b), and (c) show the horizontal, vertical, and diagonal correlation of Peppers's image before encryption, respectively. Figures 4.3 (c), (d), and (e) show the horizontal, vertical, and diagonal correlation of Peppers's image after encryption, respectively.

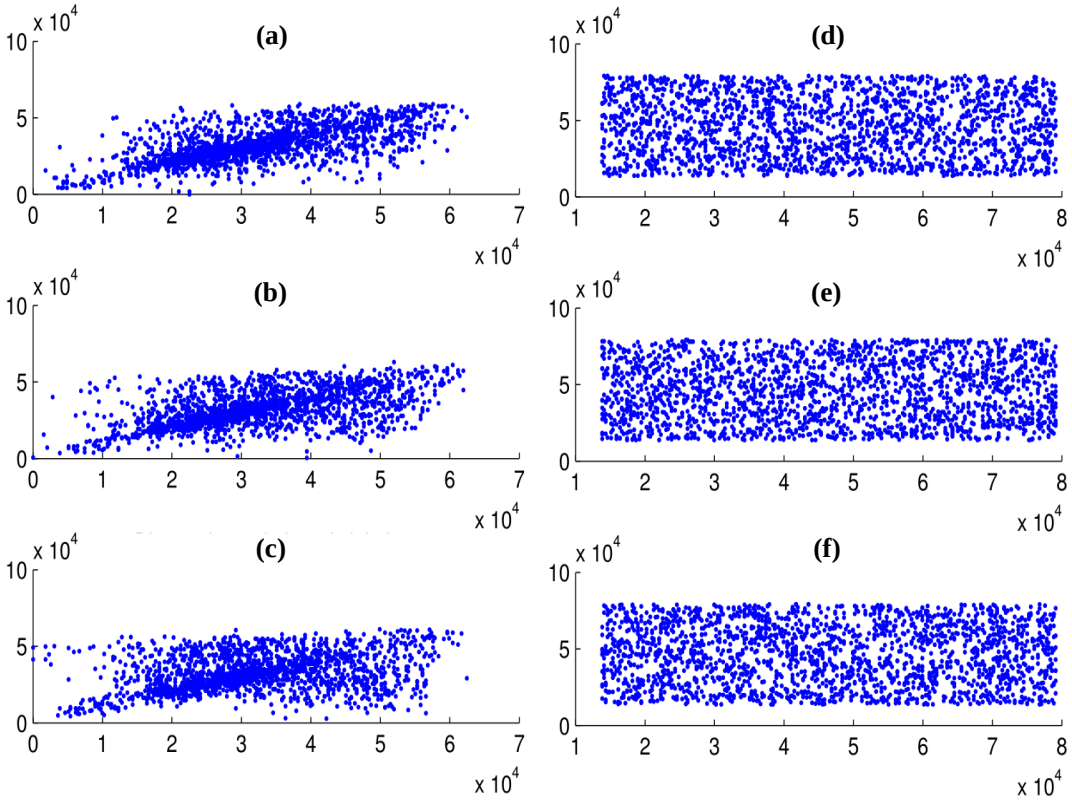



Figure 4.3: Correlation coefficients of IDN for Peppers's image before encryption (a) Horizontal, (b) Vertical, and (c) Diagonal; after encryption (d) Horizontal, (e) Vertical, and (f) Diagonal

#### 4.4.2.3 Differential analysis

To evaluate the sensitivity of IDN, two same images having a difference of one pixel are encrypted using the same security key. Then, NPCR and UACI are calculated using Eqs. (1.12) and (1.13) for each test image. Due to stochastic nature of differential evolution, the values of NPCR and UACI for IDN is evaluated on $30$ iterations. The pixel of each image is changed randomly in every iteration. The mean and standard deviation of $30$ iterations are represent in Table 4.5. From Table 4.5, it is observed that IDN achieves significant NPCR and UACI values. Therefore, the IDN provides better efficiency against differential attacks.

Table 4.5: NPCR and UACI (mean $\pm$ standard deviation) of IDN

| Image | NPCR | UACI |
|---|---|---|
| Ultrasound | $0.9960 \pm 0.0006$ | $0.3342 \pm 0.0013$ |
| Crown | $0.9965 \pm 0.0004$ | $0.3339 \pm 0.0018$ |
| Peppers | $0.9965 \pm 0.0005$ | $0.3342 \pm 0.0016$ |
| Lake | $0.9961 \pm 0.0008$ | $0.3344 \pm 0.0022$ |
| House | $0.9962 \pm 0.0005$ | $0.3345 \pm 0.0014$ |

#### 4.4.2.4 Secret key analysis

Secret keys are the core part of any encryption algorithm. The key space and sensitivity are main properties of a secret key [60].

##### 4.4.2.4.1 Secret key space

In IDN, the size of secret key $(B_k)$ is same as of input image $(I)$. The size of $B_k$ is $256 \times 256$ bytes which means the key space of $B_k$ is $256! \times 256! \approx 2^{3369}$ bytes. The key size of $B_k$ is about $(2^{3369})^8 = 2^{26952}$, *i.e.,* about $26952$ bits. Therefore, IDN has a huge key space to resist the brute-force attacks.

##### 4.4.2.4.2 Secret key sensitivity

The image encryption technique should be sensitive toward initial values of a secret key. To evaluate the sensitivity of IDN's secret key, select an input image $(I)$ and generate secret key $(B_k)$. Then, generate second secret key $(B_k{}')$ with the difference of one pixel. Thus, two encrypted images are generated by utilizing $B_k$ and $B_k{}'$. Finally, the difference between encrypted images is calculated by using Eq. (1.12). Here, $E$ and $E'$ are two ciphered images generated from same plain image using two secret keys, *i.e.,* $B_k$ and $B_k{}'$. Figure 4.4 and Table 4.6 show the difference between $E$ and $E'$. From the Table 4.6, it can be observed that the IDN generates totally different encrypted images when secret keys

differ with only single pixel. The resultant values demonstrate that IDN is highly sensitive to initial values.

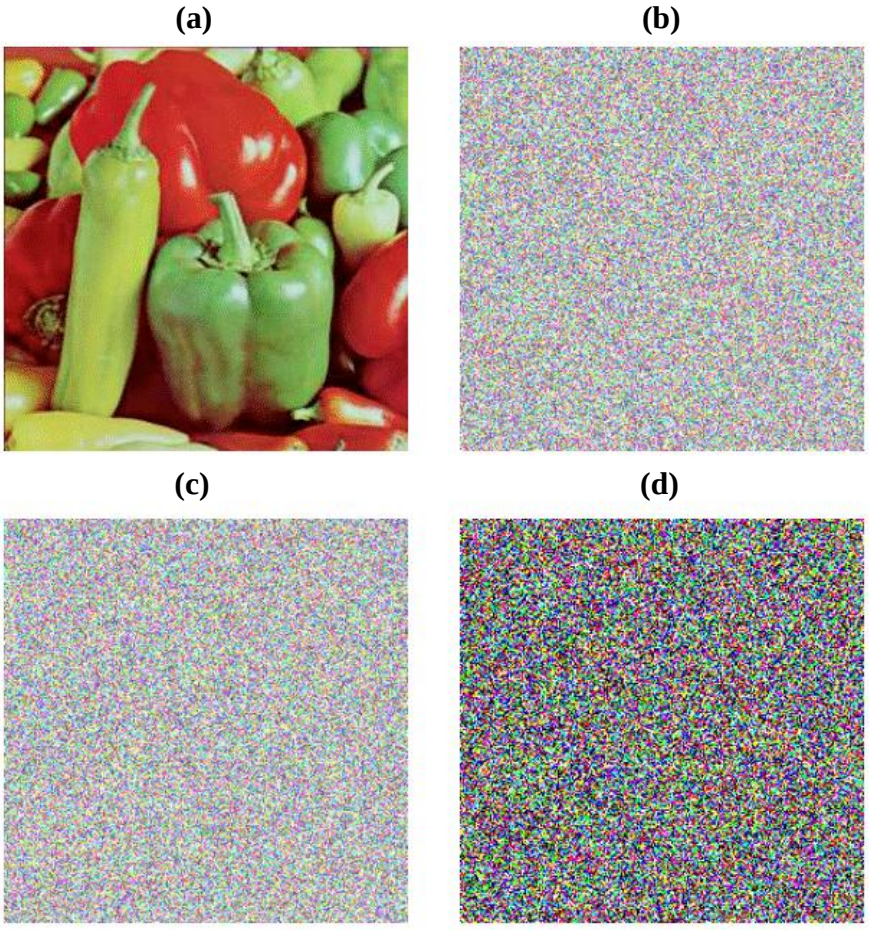



Figure 4.4: Secret key sensitivity of IDN (a) Peppers's image, (b) Encrypted image with first secret key, *i.e.,* $B_k$, (c) Encrypted image using second secret key, *i.e.,* $B_k{}'$ with the difference of one pixel, and (d) Difference between (b) and (c)

Table 4.6: Difference between $E$ and $E'$ images of IDN

| | Ultrasound | Crown | Peppers | House | Lake |
|---|---|---|---|---|---|
| Difference | 99.940 | 99.934 | 99.944 | 99.996 | 99.974 |

### 4.4.3 Comparative analysis

IDN is compared with seven well-known existing techniques such as MCIIR [139], PTS [144], IWT [145], CDWT [95], LWT [146], OSTM [176], and PSCT [54].

Table 4.7 shows the performance comparison of IDN with existing image encryption techniques in terms of entropy. It is observed that IDN provides maximum entropy as

compared to other techniques. MCIIR also provides good entropy among other techniques but not better than IDN.

Table 4.7: Comparative analysis of IDN in terms of entropy

| Technique | Ultrasound | Crown | Peppers | Lake | House |
|---|---|---|---|---|---|
| MCIIR | 7.9971 | 7.9961 | 7.9903 | 7.9974 | 7.9973 |
| PTS | 7.9484 | 7.9540 | 7.9544 | 7.9575 | 7.9500 |
| IWT | 7.9352 | 7.9339 | 7.9554 | 7.9359 | 7.9361 |
| CDWT | 7.8693 | 7.8753 | 7.8688 | 7.8719 | 7.8696 |
| LWT | 7.8739 | 7.8702 | 7.8727 | 7.8947 | 7.8756 |
| OSTM | 7.8334 | 7.9302 | 7.7721 | 7.8923 | 7.9239 |
| PSCT | 7.9634 | 7.8302 | 7.8921 | 7.9223 | 7.9439 |
| IDN | 7.9980 | 7.9982 | 7.9979 | 7.9976 | 7.9980 |

The correlation of an image is calculated as horizontal, diagonal, and vertical correlations. Table 4.8 shows the horizontal correlation of IDN in comparison with the others. IWT provides minimum correlation (*i.e.,* $-0.0137$) for Lake's image. In the case of house's image, minimum correlation (*i.e.,* $-0.0181$) is provided by CDWT. However, IDN provides minimum horizontal correlation in most of the cases. It implies that the attacker cannot extract any statistical information from the encrypted images to recover the original image.

Table 4.8: Comparative analysis of IDN with respect to horizontal correlation

| Technique | Ultrasound | Crown | Peppers | Lake | House |
|---|---|---|---|---|---|
| MCIIR | 0.0060 | 0.0018 | 0.0020 | $-0.0005$ | 0.0010 |
| PTS | 0.0137 | 0.0064 | 0.0126 | 0.0016 | 0.0053 |
| IWT | 0.0089 | 0.0149 | 0.0106 | $-0.0137$ | $-0.0173$ |
| CDWT | 0.0152 | 0.0126 | 0.0152 | $-0.0110$ | $-0.0181$ |
| LWT | 0.0108 | 0.0212 | 0.0040 | $-0.0107$ | $-0.0087$ |
| OSTM | 0.0128 | 0.0220 | 0.0190 | 0.0110 | 0.0089 |
| PSCT | 0.0099 | 0.0120 | 0.0240 | 0.0080 | 0.0077 |
| IDN | $-0.0003$ | 0.0015 | 0.0001 | 0.0022 | 0.0032 |

Table 4.9 shows the comparison of IDN with other existing techniques using diagonal correlation. IDN provides minimum diagonal correlation as compared to other existing techniques. However, in the case of House's image, LWT provides minimum correlation (*i.e.,* $-0.0211$). From the diagonal correlation of IDN, it can be seen that attacker cannot find any correlation among the adjacent pixels of an encrypted image.

Table 4.9: Comparative analysis of IDN in terms of diagonal correlation

| Technique | Ultrasound | Crown | Peppers | Lake | House |
|---|---|---|---|---|---|
| MCIIR | 0.0037 | 0.0047 | −0.0002 | 0.0037 | 0.0009 |
| PTS | 0.0067 | 0.0051 | 0.0093 | 0.0058 | 0.0064 |
| IWT | 0.0138 | 0.0019 | 0.0053 | 0.0013 | −0.0170 |
| CDWT | 0.0088 | −0.0012 | 0.0098 | 0.0092 | −0.0164 |
| LWT | 0.0068 | −0.0057 | 0.0081 | 0.0052 | −0.0211 |
| OSTM | 0.0168 | 0.0171 | 0.0091 | 0.0042 | 0.0121 |
| PSCT | 0.0159 | 0.0120 | 0.0240 | 0.0109 | 0.0198 |
| IDN | 0.0065 | 0.0042 | −0.0008 | 0.0006 | 0.0028 |

Table 4.10 shows the vertical correlation obtained from the IDN and existing techniques. However, in the case of House's image, minimum correlation is provided by LWT.

Table 4.10: Comparative analysis of IDN in terms of vertical correlation

| Technique | Ultrasound | Crown | Peppers | Lake | House |
|---|---|---|---|---|---|
| MCIIR | 0.0113 | 0.0074 | 0.0038 | 0.0047 | 0.0011 |
| PTS | 0.0136 | 0.0095 | 0.0068 | 0.0150 | −0.0137 |
| IWT | 0.0078 | 0.0148 | 0.0058 | 0.0035 | 0.0171 |
| CDWT | 0.0121 | 0.0059 | 0.0120 | 0.0095 | 0.0110 |
| LWT | 0.0150 | 0.0141 | 0.0113 | 0.0160 | −0.0145 |
| OSTM | 0.0140 | 0.0143 | 0.0133 | 0.0125 | 0.0154 |
| PSCT | 0.0178 | 0.0103 | 0.0130 | 0.0100 | 0.0165 |
| IDN | 0.0023 | −0.0048 | 0.0002 | −0.0029 | 0.0023 |

Table 4.11: Comparative analysis of IDN using NPCR

| Technique | Ultrasound | Crown | Peppers | Lake | House |
|---|---|---|---|---|---|
| MCIIR | 0.9959 | 0.9960 | 0.9962 | 0.9961 | 0.9952 |
| PTS | 0.9950 | 0.9732 | 0.9730 | 0.9734 | 0.9733 |
| IWT | 0.9953 | 0.9962 | 0.9958 | 0.9960 | 0.9959 |
| CDWT | 0.9961 | 0.9954 | 0.9959 | 0.9963 | 0.9964 |
| LWT | 0.9946 | 0.9956 | 0.9943 | 0.9959 | 0.9960 |
| OSTM | 0.9932 | 0.9942 | 0.9934 | 0.9939 | 0.9945 |
| PSCT | 0.9937 | 0.9951 | 0.9947 | 0.9940 | 0.9939 |
| IDN | 0.9966 | 0.9969 | 0.9970 | 0.9969 | 0.9967 |

Diffusion is an important property for secure encryption techniques according to Claude Shannon [177]. It means slightest modification in the original image will change the

encrypted image completely. Table 4.11 and 4.12 show the performance comparison of IDN with the existing techniques in terms of NPCR and UACI, respectively.

IDN provides better values of NPCR and UACI as compared to the other techniques. The IDN gives highest values of NPCR and UACI are $0.9970$ (*i.e.,* $99.7\%$) and $0.3366$ (*i.e.,* $33.6\%$), respectively. It means that when only one pixel is changed in an original image, then IDN generates a completely different ciphered image.

Table 4.12: Comparative analysis of IDN using UACI

| Technique | Ultrasound | Crown | Peppers | Lake | House |
|---|---|---|---|---|---|
| MCIIR | 0.3339 | 0.3314 | 0.3318 | 0.3316 | 0.3318 |
| PTS | 0.3329 | 0.3341 | 0.3347 | 0.3350 | 0.3327 |
| IWT | 0.3330 | 0.3344 | 0.3353 | 0.3347 | 0.3352 |
| CDWT | 0.3346 | 0.3347 | 0.3344 | 0.3351 | 0.3339 |
| LWT | 0.3345 | 0.3349 | 0.3351 | 0.3346 | 0.3342 |
| OSTM | 0.3332 | 0.3339 | 0.3346 | 0.3342 | 0.3340 |
| PSCT | 0.3322 | 0.3329 | 0.3331 | 0.3336 | 0.3330 |
| IDN | 0.3355 | 0.3357 | 0.3358 | 0.3366 | 0.3359 |

Table 4.13 shows MAE obtained from IDN and the existing techniques. IDN provides maximum difference between input and encrypted images. Hence, the attacker unable to find any clue of the real information from the encrypted image.

Table 4.13: Comparative analysis of IDN using MAE

| Technique | Ultrasound | Crown | Peppers | Lake | House |
|---|---|---|---|---|---|
| MCIIR | 79.2275 | 72.8957 | 69.5667 | 73.4669 | 80.1179 |
| PTS | 79.2752 | 75.0392 | 71.9020 | 75.99074 | 82.6779 |
| IWT | 79.0175 | 73.8975 | 70.1388 | 75.2574 | 82.0500 |
| CDWT | 80.9763 | 74.4053 | 70.8065 | 75.5613 | 82.7718 |
| LWT | 80.9788 | 74.3374 | 71.0397 | 75.6586 | 82.6055 |
| OSTM | 72.5894 | 76.8897 | 71.8970 | 72.4200 | 79.0023 |
| PSCT | 73.4009 | 73.9847 | 75.0184 | 72.2202 | 76.6805 |
| IDN | 86.1587 | 83.1778 | 85.4931 | 88.9809 | 92.8469 |

Table 4.14 shows the PSNR obtained from IDN and other techniques. IDN provides better PSNR in comparison to the other techniques. Therefore, IDN gives good quality decrypted images.

Table 4.14: Comparative analysis of IDN using PSNR

| Technique | Ultrasound | Crown | Peppers | Lake | House |
|---|---|---|---|---|---|
| MCIIR | 65.6628 | 65.2259 | 65.3524 | 65.1093 | 64.6272 |
| PTS | 32.6286 | 32.2669 | 32.7895 | 32.9856 | 32.5670 |
| IWT | 64.6118 | 64.3108 | 64.2486 | 64.3411 | 64.4623 |
| CDWT | 64.6482 | 64.1811 | 64.1947 | 64.2271 | 64.0407 |
| LWT | 64.9151 | 64.6722 | 64.6795 | 64.6728 | 64.6423 |
| OSTM | 65.1151 | 65.7212 | 65.1975 | 65.6998 | 65.2560 |
| PSCT | 66.2895 | 66.5122 | 66.7009 | 66.0891 | 66.4780 |
| IDN | 80.3210 | 87.4729 | 89.3253 | 89.2722 | 84.9844 |

## 4.5 Summary

An image encryption technique based on differential evolution (IDN) has been proposed in this chapter. IDN used Arnold transform to produce the scrambled image. Thereafter, NSCT decomposed the scrambled image into sub-bands. The tuned parameters are generated by differential evolution, which are used in beta chaotic map to generate a secret key. The coefficients of sub-bands are encrypted using the secret key. The mean improvement of IDN in terms of entropy, NPCR, UACI, $PSNR$ and $MAE$ are $0.22$ %, $0.09$ %, $0.10$ %, $20.5$ % (dB), and $9.9$ %, respectively. The mean reduction of IDN in case of a correlation coefficient is $2.9$ %. Therefore, the results reveal that IDN outperforms the others.

# Chapter 5

# Memetic differential evolution based image encryption

This chapter proposes an efficient Image encryption technique based on Intertwining logistic map, memetic differential evolution, and Arnold transform known as IIMA. Memetic differential evolution uses a local search mechanism to improve the solutions obtained by differential evolution. The performance of IIMA is also compared with IDN.

## 5.1 Proposed image encryption technique

Image encryption technique based on intertwining logistic map, memetic differential evolution, and Arnold transform known as IIMA is proposed. It contains encryption and decryption processes.

### 5.1.1 Encryption process

Figure 5.1 depicts the encryption process of IIMA. Initially, an input color image is decomposed into three channels *i.e.,* red ($I_R$), green ($I_G$), and blue ($I_B$). The pixels position of these channels is permuted using Arnold transform. Thereafter, the secret keys are developed by intertwining logistic map based on memetic differential evolution to encrypt the channels. All encrypted channels are combined to produce the final encrypted image. The following are the various steps required to implement IIMA.

**Step 1:** An input color image ($I$) with rows ($w$ ) and columns ($h$).

**Step 2:** Decompose $I$ into $I_R$, $I_G$, and $I_B$.

**Step 3:** Apply Arnold transform on $I_R$, $I_G$, and $I_B$ to generate permuted channels, *i.e.,* $I'_R$, $I'_G$, and $I'_B$. It is computed as follows:

$$I'_R(\chi', \varphi')^{it} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} I_R(\chi, \varphi)^{it-1} (mod \ \ S_z(I_R)) \tag{5.1}$$

$$I'_G(\chi', \varphi')^{it} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} I_G(\chi, \varphi)^{it-1} (mod \ \ S_z(I_G)) \tag{5.2}$$

$$I'_B(\chi', \varphi')^{it} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} I_B(\chi, \varphi)^{it-1} (mod \ \ S_z(I_B)) \tag{5.3}$$

Here, $it = 1, 2, 3, \ldots, n$ represents the number of iterations. $S_z$ represents size of channels. $(\chi, \varphi)$ and $(\chi', \varphi')$ represent actual and permutated co-ordinates of an image.

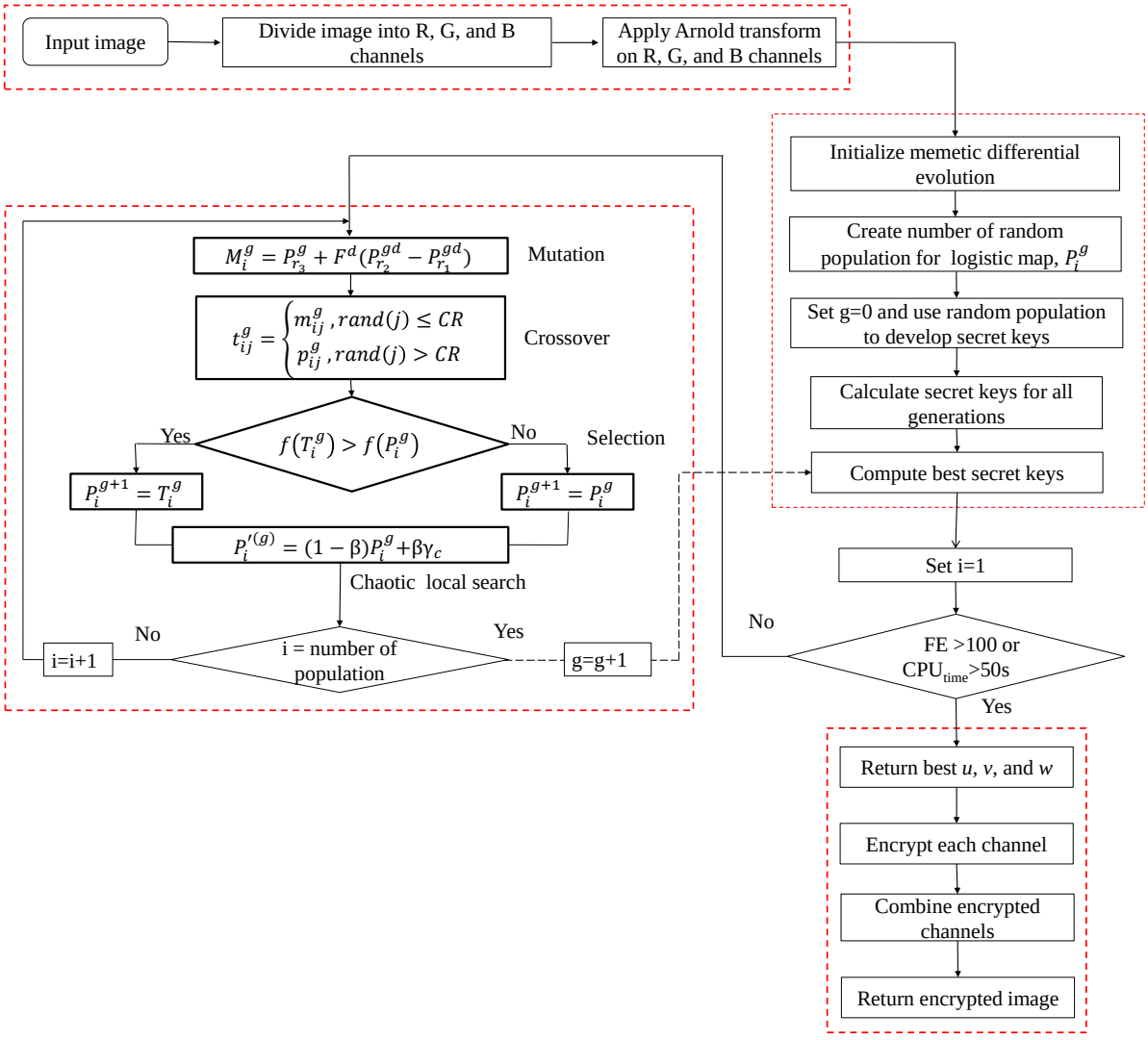



Figure 5.1: Flowchart of an image encryption technique based on intertwining logistic map, memetic differential evolution, and Arnold transform

**Step 4:** Generate the secret keys, *i.e.*, $x$, $y$, and $z$ using intertwining logistic map (see Eq. (1.8)). Memetic differential evolution is utilized to optimize the parameters needed by intertwining logistic map. The size of $x$, $y$, and $z$ is same as the size of channels.

**Step 4.1 Population initialization:** The initial solutions of memetic differential evolution are defined as:

$$P_i^g = [p_{i1}^g, p_{i2}^g, \ldots, p_{i8}^g], \;\; i \in \{1, 2, \ldots, N_p\} \tag{5.4}$$

where $p_{ij}$ represents $j^{th}$ parameter from $i^{th}$ solution in $g$ generation. $N_p$ denotes the population size.

In this step, each solution is generated randomly using normal distribution. The size of each random solution ($P_i^g$) is $8$. The first seven parameters of each $P_i^g$ are assigned to intertwining logistic map for developing the secret keys and last parameter is considered as encryption factor ($\alpha_e$). Every $P_i^g$ selects scaling factor ($F$) and crossover rate ($CR$) independently within the range of $[0, 1]$. The fitness function of encrypted images, generated through random $P_i^g$ solutions, is evaluated using Eq. (5.5).

**Step 4.2 Fitness function computation:** Entropy and correlation coefficients are used for evaluating the fitness of solutions obtained from memetic differential evolution. The main objective of memetic differential evolution is to maximize the entropy and minimize correlation coefficient of an encrypted image.

The fitness function ($f(s_1, s_2)$) is subjected to two constraints. It is computed as

$$\begin{aligned} &\text{optimize} \;\; f(s_1, s_2) \\ &\text{subjec to} \;\; s_1 > 7.9, s_2 \in \{-0.05, 0.05\} \end{aligned} \tag{5.5}$$

Here, $s_1$ denotes entropy which can be computed using Eq. (1.19). $s_2$ represents correlation and can be calculated using Eqs. (1.15)-(1.18).

**Step 4.3 Mutation:** Memetic differential evolution generates a mutated solution ($M_i^g$) by summing the difference of two solutions into a third one.

$$\begin{aligned} M_i^g &= P_{r_3}^g + F^d \cdot (P_{r_2}^{gd} - P_{r_1}^{gd}) \\ r_1, r_2, r_3 &\in \{1, 2, \ldots, N_p\}, \;\; d \in \{1, 2\} \end{aligned} \tag{5.6}$$

where $r_1$, $r_2$, and $r_3$ are randomly selected solutions from the same population (but $r_1 \neq r_2 \neq r_3 \neq i$). $d$ represents the number of difference solutions participating in mutation process. $F$ is scaling factor and lies in the range of $[0, 1]$.

**Step 4.4 Crossover:** The trial solution (*i.e.,* $T_i^g$) is generated from the combination of mutated solution ($M_i^g$) and its initial solution ($P_i^g$). The parameters of $T_i^g$ are selected as:

$$t_{ij}^g = \begin{cases} m_{ij}^g, & rand(j) \leq CR, \\ p_{ij}^g, & rand(j) > CR. \end{cases} \tag{5.7}$$

Here, $j \in \{1, 2, \ldots, 8\}$. $t_{ij}^g$ and $m_{ij}^g$ are the parameters of $T_i^g$ and $M_i^g$, respectively. $CR$ denotes the crossover rate.

**Step 4.5 Selection:** The parameters of each $T_i^g$ are assigned to intertwining logistic map to obtain the secret keys. These keys are used to generate $N_p$ encrypted images.

In selection process, the fitness of encrypted images generated through $T_i^g$ and old $P_i^g$ are compared. The better solution will survive for next generation.

$$P_i^{g+1} = \begin{cases} T_i^g, & \text{if } f(T_i^g) > f(P_i^g), \\ P_i^g, & \text{otherwise.} \end{cases} \tag{5.8}$$

**Step 4.6 Chaotic local search:** The best solution, *i.e.,* $P_i^g$ is further refined by applying Chaotic linear search (CLS). CLS is computed as:

$$P_i^{'(g)} = (1 - \beta)P_i^g + \beta\gamma_c \tag{5.9}$$

where $P_i^{'(g)}$ represents new solution of $P_i^g$ generated by CLS. $\beta$ denotes the shrinking scale and is calculated as:

$$\beta = 1 - \left| \frac{FEs - 1}{FEs} \right|^{\lambda} \tag{5.10}$$

Here, $FEs$ represents the current function evaluations. The shrinking speed is controlled by $\lambda$. If the value of $\lambda$ is low, then shrinking speed is high.

$\gamma_c$ is computed as:

$$\gamma_c = A + \gamma_j^{\nu} \cdot (B - A), \tag{5.11}$$

where $[A, B]$ represents the search space of $P_i$. $\gamma_j^{\nu}$ is obtained from chaotic logistic map. Chaotic logistic map is evaluated as:

$$\gamma_j^{\nu+1} = \mu\gamma_j^{\nu}(1 - \gamma_j^{\nu}), \;\; \nu = 1, 2, \ldots; \;\; \gamma_j \in (0, 1), \tag{5.12}$$

where $\gamma_j^{\nu}$ is the $j^{th}$ cahaotic parameter in $\nu^{th}$ generation and $\gamma_j \neq 0.25$, $0.5$, and $0.75$. $\mu$ is the bifurcation control parameter.

**Step 4.7 Termination criteria:** The two conditions such as CPU time and function evaluations are taken into consideration for algorithm termination. As known in prior, the higher value of function evaluations provide more relevant results. However, in some cases, it takes too much CPU time. So, it becomes bottleneck of the algorithm. Therefore, the following termination criteria is proposed to terminate the algorithm.

$$Termination = \begin{cases} 1, & FE > 100, \\ 1, & CPU_{time} > 50sec, \\ 0, & otherwise. \end{cases} \tag{5.13}$$

Here, $CPU_{time}$ represents the overall execution time spent by memetic differential evolu-

tion so far.

If termination criteria is not satisfied, then the Steps $4.3$ - $4.7$ are repeated. Otherwise, the optimal parameters are returned.

**Step 5:** Three secret keys, *i.e.,* $x_i$, $y_i$, and $z_i$ (where $i = \{1, 2, \ldots, w \times h\}$) and encryption factor ($\alpha_e$) obtained from memetic differential evolution are used to encrypt $I'_R$, $I'_G$, and $I'_B$ channels.

$$ER = \mod (\alpha_e \times I'_R + (1 - \alpha_e) \times x, p_k) \tag{5.14}$$

$$EG = \mod (\alpha_e \times I'_G + (1 - \alpha_e) \times y, p_k) \tag{5.15}$$

$$EB = \mod (\alpha_e \times I'_B + (1 - \alpha_e) \times z, p_k) \tag{5.16}$$

Here, $ER$, $EG$, and $EB$ represent the encrypted red, green, and blue channels, respectively. $p_k$ denotes the peak pixel value of $I$.

**Step 6:** The encrypted channels $ER$, $EG$, and $EB$ are concatenated to evaluate the encrypted image ($E$) as:

$$E = cat(ER, EG, EB) \tag{5.17}$$

### 5.1.2 Decryption process

To decrypt the encrypted image, the same secret keys (*i.e.,* $x$, $y$, and $z$) and same encryption factor ($\alpha_e$) are required. Therefore, the tuned parameters are required to be communicated with receiver.

**Step 1:** $E$ is decomposed into three encrypted color channels, namely red ($ER$), green ($EG$), and blue ($EB$).

**Step 2:** The secret keys, *i.e.,* $x$, $y$, and $z$ are generated by utilizing tunned parameters to intertwining logistic map.

**Step 3:** Apply $x$, $y$, $z$, and $\alpha_e$ on each encrypted channel to obtain decrypted channels as follows:

$$I'_R = (ER - (1 - \alpha_e) \times x)/\alpha_e \tag{5.18}$$

$$I'_G = (EG - (1 - \alpha_e) \times y)/\alpha_e \tag{5.19}$$

$$I'_B = (EB - (1 - \alpha_e) \times z)/\alpha_e \tag{5.20}$$

Here, $I'_R$, $I'_G$, and $I'_B$ represent decrypted color channels.

**Step 4:** The original channels *i.e.,* $I_R$, $I_G$, and $I_B$ are obtained using the inverse of

Arnold transform on $I'_R$, $I'_G$, and $I'_B$, respectively.

$$I_R(\chi', \varphi')^{it} = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix} I'_R(\chi, \varphi)^{it-1}(mod \ \ S_z(I'_R)) \tag{5.21}$$

$$I_G(\chi', \varphi')^{it} = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix} I'_G(\chi, \varphi)^{it-1}(mod \ \ S_z(I'_G)) \tag{5.22}$$

$$I_B(\chi', \varphi')^{it} = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix} I'_B(\chi, \varphi)^{it-1}(mod \ \ S_z(I'_B)) \tag{5.23}$$

**Step 5:** The final decrypted image ($D$) is obtained by combining $I_R$, $I_G$, and $I_B$ channels.

$$D = cat(I_R, I_G, I_B) \tag{5.24}$$

where $cat$ represents concatenation operation.

## 5.2 Experimental results and discussion

To evaluate the effectiveness of IIMA, it is compared with IWT [145], CDWT [95], LWT [146], OSTM [176], PSCT [54], IGN, and IDN. IIMA is tested on five benchmark color images of size $256 \times 256$. These are taken from [167].

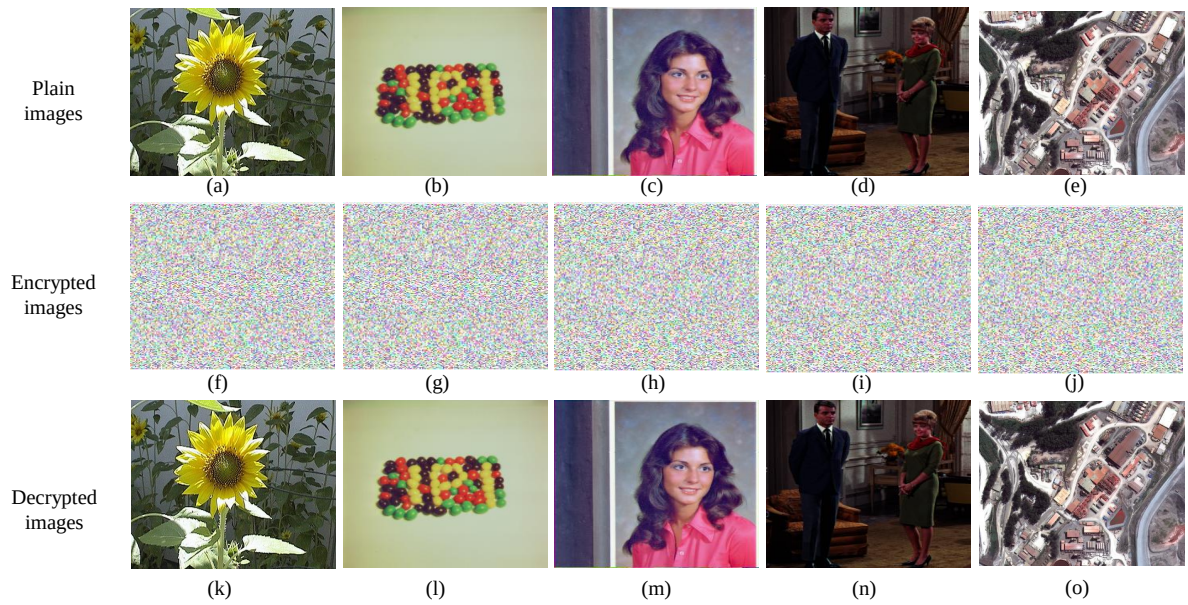



Figure 5.2: Performance evaluation of IIMA (a)-(e) Plain images, (f)-(j) Encrypted images, and (k)-(o) Decrypted images

Figures 5.2 (a)-(e) show the plain color images. Figures 5.2 (f)-(j) show the encrypted images which are generated by the encryption process of IIMA. The decrypted images

which are obtained through decryption process can be seen from Figures 5.2 (k)-(o). The input and decrypted images are identical to each other. Therefore, IIMA provides a significant visual quality.

### 5.2.1 Performance evaluation

Performance of IIMA is evaluated by considering various performance measures such as entropy and peak signal noise ratio.

#### 5.2.1.1 Entropy

Entropy is an important parameter to check the degree of randomness. Table 5.1 shows the performance comparison between IIMA and existing image encryption techniques in terms of entropy. It can be seen from the table that the entropy values of IIMA is closed to ideal value (*i.e.,* $8$). The mean improvement is observed in IIMA with respect to entropy over IWT, CDWT, LWT, OSTM, PSCT, IGN, and IDN is 0.135%, 0.149%, 0.141%, 0.143%, 0.089%, 0.051%, and 0.048%, respectively. Thus, IIMA has maximum entropy as compared to other techniques.

Table 5.1: Comparison of IIMA in terms of entropy (in bit/pixel)

| Technique | Sunflower | Beans | Women | Couple | Remote |
|---|---|---|---|---|---|
| IWT | 7.9853 | 7.9830 | 7.9855 | 7.9850 | 7.9740 |
| CDWT | 7.9694 | 7.9754 | 7.9789 | 7.9710 | 7.9797 |
| LWT | 7.9739 | 7.9703 | 7.9728 | 7.9748 | 7.9742 |
| OSTM | 7.9734 | 7.9725 | 7.9752 | 7.9744 | 7.9732 |
| PSCT | 7.9941 | 7.9953 | 7.9948 | 7.9924 | 7.9551 |
| IGN | 7.9970 | 7.9862 | 7.9969 | 7.9971 | 7.9974 |
| IDN | 7.9972 | 7.9970 | 7.9971 | 7.9976 | 7.9975 |
| **IIMA** | 7.9989 | 7.9987 | 7.9988 | 7.9990 | 7.9984 |

#### 5.2.1.2 Peak signal to noise ratio

Table 5.2 shows PSNR obtained from IIMA and the existing image encryption techniques. IIMA shows mean improvement in terms of PSNR over competitive techniques such as IWT, CDWT, LWT, OSTM, PSCT, IGN, and IDN by 0.17.12%, 13.42%, 9.97%, 12.47%, 11.23%, 5.97%, and 2.17%, respectively. It indicates that IIMA has better PSNR value than the existing techniques.

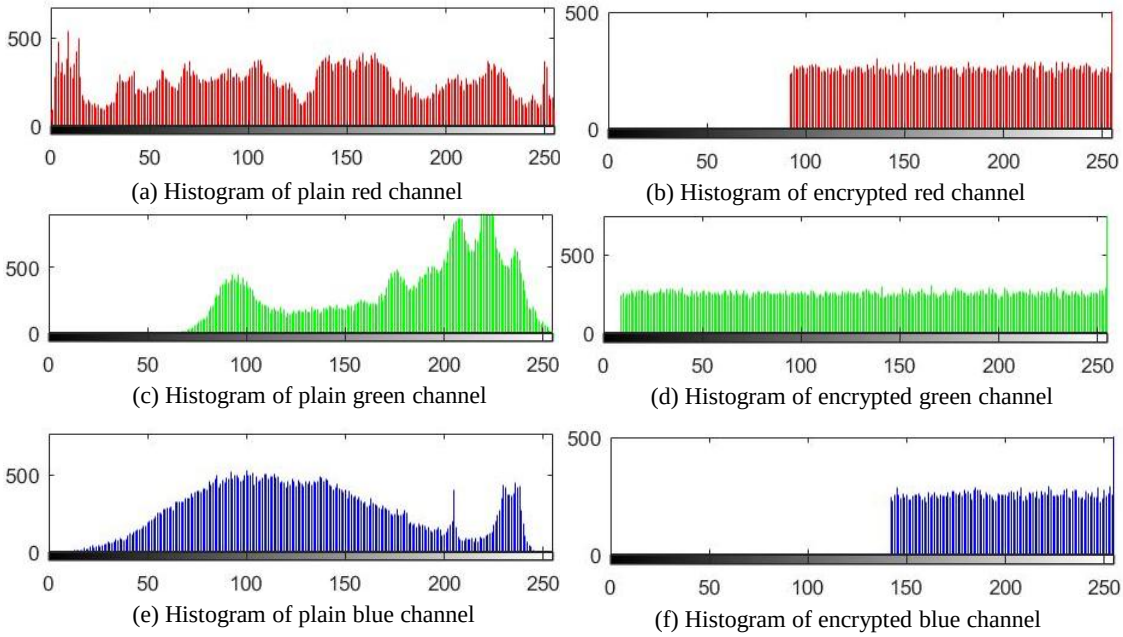
Table 5.2: Comparative analysis of IIMA using PSNR (in dB)

| Technique | Sunflower | Beans | Women | Couple | Remote |
|---|---|---|---|---|---|
| IWT | 63.6115 | 63.4105 | 63.2356 | 63.4311 | 63.5113 |
| CDWT | 67.6352 | 67.1511 | 67.1739 | 68.2291 | 67.4321 |
| LWT | 70.7181 | 71.6922 | 70.6978 | 73.6925 | 71.7601 |
| OSTM | 68.1181 | 68.9212 | 68.1798 | 68.6775 | 68.3491 |
| PSCT | 69.2578 | 69.8122 | 69.9007 | 69.0571 | 69.0798 |
| IGN | 74.6256 | 74.2667 | 74.9578 | 74.7586 | 74.7501 |
| IDN | 78.6625 | 78.2287 | 78.4823 | 78.1074 | 77.4590 |
| **IIMA** | 80.6806 | 86.8489 | 83.4284 | 82.2922 | 87.5332 |

### 5.2.2 Security analysis

The effectiveness of IIMA is tested against various attacks by considering statistical, differential, and secret key analyses.

#### 5.2.2.1 Histogram analysis




(a) Histogram of plain red channel (b) Histogram of encrypted red channel

(c) Histogram of plain green channel (d) Histogram of encrypted green channel

(e) Histogram of plain blue channel (f) Histogram of encrypted blue channel

Figure 5.3: Histograms of Sunflower image (a) Plain red channel, (b) Encrypted red channel, (c) Plain green channel, (d) Encrypted green channel, (e) Plain blue channel, and (f) Encrypted blue channel

The histograms show the intensity of each pixel value of an image. The statistical information can be leaked out through histograms [1, 5]. Hence, it is required that the

histogram of an encrypted image should be uniformly distributed. Figure 5.3 shows the histograms of $R$, $G$, and $B$ channels of plain and encrypted Sunflower image. It can be observed from Figure 5.3 that the histograms of encrypted channels are quite different from the histograms of plain channels. The histograms of encrypted channels are uniformly distributed. Therefore, IIMA is secured against statistical attacks.

#### 5.2.2.2 Correlation coefficient

Tables 5.3, 5.4, and 5.5 depict the horizontal, diagonal, and vertical correlations comparison between IIMA and the well-known existing image encryption techniques, respectively. In these tables, the horizontal, diagonal, and vertical correlations of IIMA are computed by considering the average of three channels, *i.e.,* $R$, $G$, and $B$. It can be seen from tables that IIMA has minimum horizontal, diagonal, and vertical correlations in maximum cases. Therefore, IIMA is secure against statistical attacks.

Table 5.3: Comparative analysis of IIMA in terms of horizontal correlation

| Technique | Sunflower | Beans | Women | Couple | Remote |
|---|---|---|---|---|---|
| IWT | 0.0150 | 0.0106 | 0.0066 | 0.0143 | −0.0137 |
| CDWT | 0.0080 | 0.0048 | 0.0152 | −0.0076 | 0.0172 |
| LWT | 0.0200 | 0.0119 | 0.0141 | −0.0162 | −0.0027 |
| OSTM | 0.0130 | 0.0218 | 0.0188 | 0.0150 | 0.0086 |
| PSCT | 0.0054 | 0.0144 | 0.0122 | 0.0072 | 0.0078 |
| IGN | 0.0014 | 0.0045 | 0.0046 | 0.0005 | 0.0040 |
| IDN | 0.0012 | 0.0018 | 0.0056 | 0.0063 | 0.0040 |
| **IIMA** | 0.0010 | 0.0025 | 0.0009 | −0.0038 | −0.0120 |

Table 5.4: Comparative analysis of IIMA with respect to diagonal correlation

| Technique | Sunflower | Beans | Women | Couple | Remote |
|---|---|---|---|---|---|
| IWT | 0.0248 | 0.0029 | 0.0074 | 0.0024 | −0.0268 |
| CDWT | 0.0088 | −0.0022 | 0.0098 | 0.0092 | 0.0242 |
| LWT | 0.0058 | −0.0077 | 0.0082 | 0.0072 | −0.0210 |
| OSTM | 0.0258 | 0.0272 | 0.0092 | 0.0032 | 0.0122 |
| PSCT | 0.0279 | 0.0220 | 0.0230 | 0.0209 | 0.0278 |
| IGN | 0.0047 | 0.0037 | −0.0002 | 0.0047 | 0.0008 |
| IDN | 0.0057 | 0.0072 | 0.0094 | 0.0078 | 0.0063 |
| **IIMA** | 0.0045 | −0.0054 | −0.0008 | 0.00015 | 0.0026 |

Table 5.5: Comparison of IIMA in terms of vertical correlation

| Technique | Sunflower | Beans | Women | Couple | Remote |
|---|---|---|---|---|---|
| IWT | 0.0068 | 0.0138 | 0.0048 | 0.0034 | 0.0159 |
| CDWT | 0.0111 | 0.0040 | 0.0110 | 0.0004 | 0.01663 |
| LWT | 0.0140 | 0.0131 | 0.0113 | 0.0160 | −0.0020 |
| OSTM | 0.0130 | 0.0129 | 0.0133 | 0.0114 | 0.0139 |
| PSCT | 0.0135 | 0.0103 | 0.0159 | 0.0100 | 0.0160 |
| IGN | 0.0113 | 0.0063 | 0.0038 | 0.0036 | 0.0010 |
| IDN | 0.0136 | 0.0016 | 0.0120 | 0.0140 | 0.0108 |
| **IIMA** | 0.0063 | −0.0050 | 0.0001 | −0.0010 | 0.0014 |

Figures 5.4 (a)-(c) show correlation coefficient (*i.e.,* horizontal, vertical, and diagonal) of red channel of input Sunflower image. Figures 5.4 (d)-(f) show correlation of encrypted red channel of Sunflower image. It indicates that the encrypted image is totally random in nature.

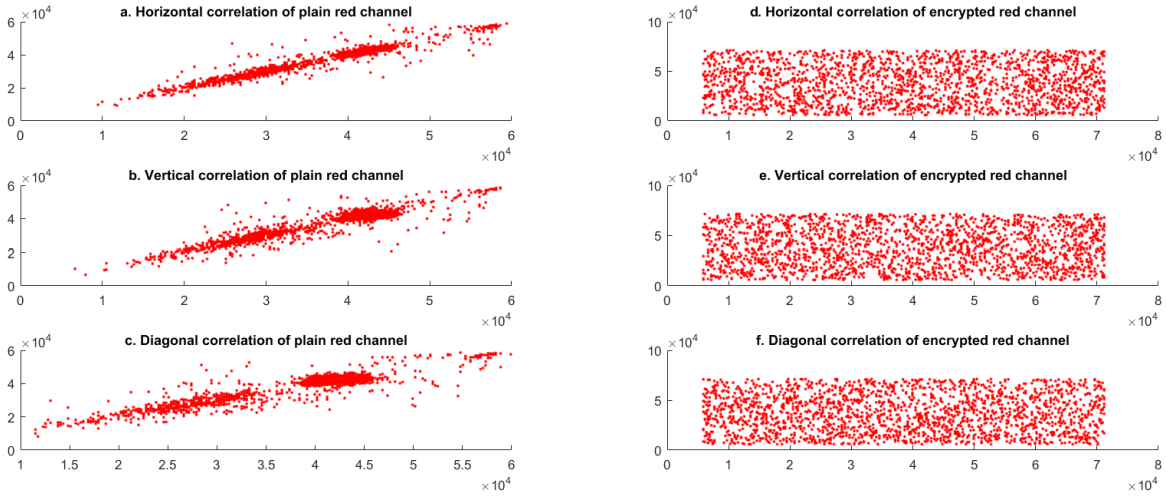



Figure 5.4: Correlation analysis of IIMA for Sunflower image: (a) Horizontal, (b) Vertical, and (c) Diagonal correlation before encryption, (d) Horizontal, (e) Vertical, and (f) Diagonal correlation after encryption

#### 5.2.2.3 Differential analysis

The sensitivity towards smallest changes of IIMA is evaluated using differential analysis [54]. Tables 5.6 and 5.7 show the performance comparison of IIMA and existing image encryption techniques based on NPCR and UACI, respectively. From Table 5.6, it has been observed that the mean improvement of IIMA in terms of NPCR over IWT, CDWT, LWT, OSTM, PSCT, IGN, and IDN is 0.09, 0.07%, 0.23%, 0.11%, 0.13%, 0.065%, and 0.12%, respectively.

From Table 5.7, the mean improvement is observed in IIMA with respect to UACI over IWT, CDWT, LWT, OSTM, PSCT, IGN, and IDN is 0.156%, 0.097%, 0.131%, 0.179%, 0.192%, 0.127%, and 0.117%, respectively. Therefore, IIMA ables to prevent the differential attacks.

Table 5.6: Comparison analysis of IIMA using NPCR (in %)

| Technique | Sunflower | Beans | Women | Couple | Remote |
|---|---|---|---|---|---|
| IWT | 99.5468 | 99.5270 | 99.5739 | 99.5943 | 99.5840 |
| CDWT | 99.6149 | 99.5438 | 99.5960 | 99.6312 | 99.6502 |
| LWT | 99.4609 | 99.5617 | 99.4398 | 99.5999 | 99.6490 |
| OSTM | 99.5226 | 99.5249 | 99.5472 | 99.5466 | 99.3555 |
| PSCT | 99.5049 | 99.5095 | 99.5359 | 99.5420 | 99.6139 |
| IGN | 99.5823 | 99.5902 | 99.6147 | 99.6098 | 99.5428 |
| IDN | 99.5120 | 99.5308 | 99.5630 | 99.5434 | 99.5709 |
| **IIMA** | 99.6300 | 99.6553 | 99.6714 | 99.6475 | 99.6689 |

Table 5.7: Comparative analysis of IIMA using UACI (in %)

| Technique | Sunflower | Beans | Women | Couple | Remote |
|---|---|---|---|---|---|
| IWT | 33.3605 | 33.4422 | 33.5370 | 33.4700 | 33.3230 |
| CDWT | 33.4623 | 33.4726 | 33.4436 | 33.5142 | 33.3918 |
| LWT | 33.4034 | 33.4021 | 33.4035 | 33.4046 | 33.4252 |
| OSTM | 33.3273 | 33.3962 | 33.4508 | 33.4511 | 33.4506 |
| PSCT | 33.2966 | 33.2887 | 33.3273 | 33.3605 | 33.3670 |
| IGN | 33.3928 | 33.4156 | 33.4863 | 33.4698 | 33.4824 |
| IDN | 33.4362 | 33.4151 | 33.4764 | 33.5027 | 33.4760 |
| **IIMA** | 33.5286 | 33.5517 | 33.5582 | 33.5634 | 33.5443 |

#### 5.2.2.4 Secret key space

The large key space is important for an image encryption technique to make brute-force attacks infeasible. In IIMA, the initial values of an intertwining logistic map generated from memetic differential evolution are used as secret keys. There are seven keys such as $x_0$, $y_0$, $z_0$, $a_1$, $a_2$, $a_3$, and $\alpha$. The key space of IIMA is $10^{98}$, if precision is set to $10^{-14}$. This key space is large enough to resist against brute-force attack.

#### 5.2.2.5 Secret key sensitivity

The sensitivity of IIMA towards secret keys has been evaluated. The initial parameters such as $x_0$, $y_0'$, $z_0'$, $a_1$, $a_2$, $a_3$, and $\alpha$ are required to generate secret keys, *i.e.,* $x$, $y$, and $z$ by using intertwining logistic map. $I_R$, $I_G$, and $I_B$ channels of $I$ are encrypted using $x$, $y$, and

$z$, respectively. The encrypted channels are combined to obtain the encrypted image, *i.e.,* $E$. Figure 5.5 (b) shows the encrypted image using original secret keys.

The minor changes are made in the initial parameter, *i.e.,* $x_0$, then different secret keys such as $x'$, $y'$, and $z'$ are generated. These modified secret keys are used to encrypt $I_R$, $I_G$, and $I_B$ channels of same input image. An encrypted image, *i.e.,* $E'$ is obtained from the combination of encrypted channels. Figure 5.5 (c) shows the encrypted image using modified secret keys. Figure 5.5 (d) shows the difference between two encrypted images, *i.e.,* $E$ and $E'$ which has small difference in initial values. Figure 5.5 (e) shows the decrypted image of Figure 5.5 (b) using original secret keys. Figure 5.5 (f) shows the decrypted image which is obtained by applying the modified secret keys on original encrypted image. It is observed that even small difference between the initial values of secret keys cannot recover the original image. The quantitative difference between $E$ and $E'$ of IIMA is shown in Table 5.8. It can be observed that IIMA provides completely different encrypted images even there is small changes in the initial conditions of secret keys.

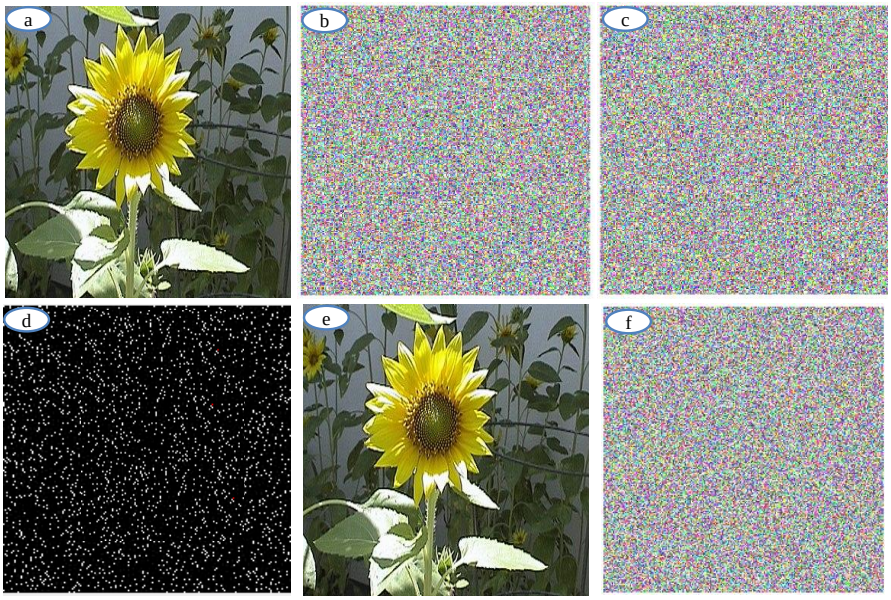



Figure 5.5: Secret key sensitivity of IIMA (a) Plain Sunflower image, (b) Encrypted image with original secret keys, (c) Encrypted image using modified secret keys, (d) Difference between (b) and (c), (e) decrypted image with original key, and (f) Decrypted image with modified keys

Table 5.8: Difference between $E$ and $E'$ images of IIMA (in %)

| | Sunflower | Beans | Women | Couple | Remote |
|---|---|---|---|---|---|
| Difference | 99.9492 | 99.9698 | 99.9584 | 99.9863 | 99.9832 |

## 5.3 Summary

A novel image encryption technique based on intertwining logistic map, memetic differential evolution, and Arnold transform (IIMA) is proposed. IIMA has the ability to overcome the problem of parameter tuning associated with intertwining logistic map. IIMA uses memetic differential evolution to resolve this issue. The effectiveness of IIMA has been tested on five well-known color images. From results, the mean improvement has been observed in IIMA over the other techniques. The parameters of IIMA such as entropy, NPCR, UACI, and PSNR have been improved by 0.051 %, 0.065 %, 0.097 %, and 2.17 % (dB), respectively. The correlation coefficient of IIMA has reduced by 0.8 %. The results reveal that IIMA provides higher efficiency and security as compared to the existing techniques.

# Chapter 6

# Parallel adaptive differential evolution based image encryption

## 6.1 Introduction

In this chapter, an efficient image encryption technique based on Secure hash algorithm (SHA-3), Adaptive differential evolution (ADE), and Lorenz-like chaotic system (ISAL) is developed. Lorenz-like chaotic system is selected because of their varying bifurcation parameter that provides complex behavior of pseudo-random numbers than the existing chaotic systems. However, it is observed that Lorenz-like chaotic system suffers from sensitivity towards the input image and parameter tuning. To overcome these issues, two techniques namely ADE and SHA-3 are used. ADE is used to optimize the required parameters of Lorenz-like chaotic system. SHA-3 is utilized to generate a 256-bit external secret key based on the input image. The optimized parameters and external secret keys are used to generate initial values for Lorenz-like chaotic system.

## 6.2 Secure hash algorithm (SHA)-3

Secure hash algorithm (SHA-3) is a member of Keccak family and follows the sponge construction. SHA-3 contains two phases such as pre-processing and inner Keccak. The pre-processing phase uses padding to evenly divide the message into r-bit blocks. The inner Keccak phase is decomposed into two main sub-phases. These are absorbing and squeezing phases. The internal state $(b)$ is used in both sub-phases and contains $1600$ bits. $b$ is divided into two parts such as rate $(r)$ (also known as block-size) that depends on the output size and capacity $(c)$. If the output size of SHA-3 is $256$, then $r$ is equal to $1152$ and $c$ is equal to $448$. The security of SHA-3 depends upon the capacity part of state. The

maximum security level is half of the capacity. Table 6.1 shows the different outputs of SHA-3 and their respective rate and capacity.

In absorbing sub-phase, the message blocks are XORed with r-bits of $b$ and the whole output is given to permutation function $f$. The function $f$ transforms the given input using $and$, $xor$, and $not$ operations. Each function contains $24$ rounds and each round further contains five sub-functions. In squeezing sub-phase, output blocks are read from the same subset of $b$ and transformed with function $f$. Figure 6.1 shows the sponge construction of SHA-3.

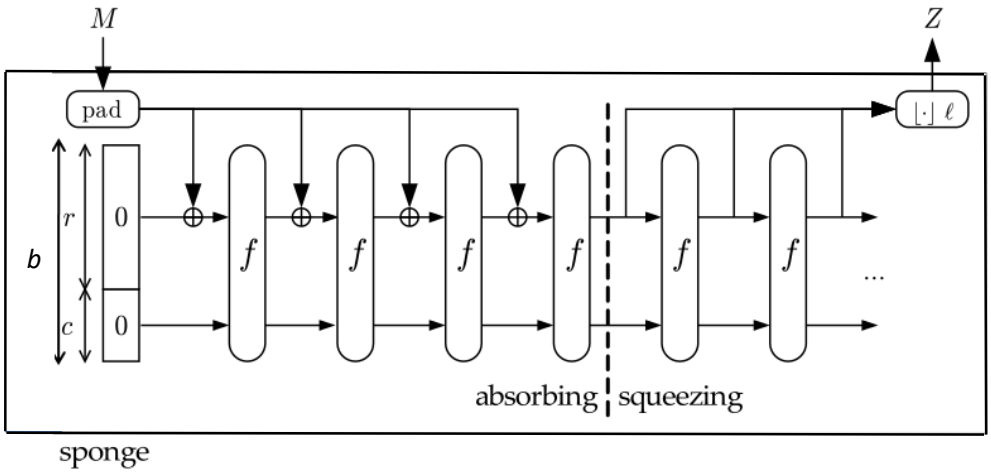



Figure 6.1: Sponge construction of SHA-3

Table 6.1: Various outputs of SHA-3 with their respective block-size and capacity

| Output (in bits) | State $(b)$ (in bits) | Rate $(r)$ (in bits) | Capacity $(c)$ (in bits) |
|---|---|---|---|
| 224 | 1600 | 1152 | 448 |
| 256 | 1600 | 1088 | 512 |
| 384 | 1600 | 832 | 768 |
| 512 | 1600 | 576 | 1024 |

## 6.3 Proposed image encryption technique

Figure 6.2 shows the flowchart of proposed Image encryption technique based on SHA-3, ADE, and Lorenz-like chaotic system (ISAL). Initially, an input image $(I)$ is decomposed into three color channels such as red $(I_R)$, green $(I_G)$ , and blue $(I_B)$. Thereafter, Adaptive differential evolution with multi-objective immune algorithm (ADE-MOIA) is utilized to optimize the parameters of Lorenz-chaotic system (see Algorithm 8). Then, the optimized parameters are used to generate secret keys such as $x$, $y$, and $z$ by using Algorithm 10. Three other keys such as $x'$, $y'$, and $z'$ are generated from the combination of $x$, $y$, and $z$.

$I_R$, $I_G$, and $I_B$ channels are diffused using $x$, $y$, and $z$. Then, the diffused channels, *i.e.,* $I'_R$ , $I'_G$, and $I'_B$ are encrypted using $x'$, $y'$, and $z'$. Finally, the encrypted channels, *i.e.,* $E_R$, $E_G$, and $E_B$ are combined to obtain the encrypted image. The complete encryption process is demonstrated in Algorithm 7.

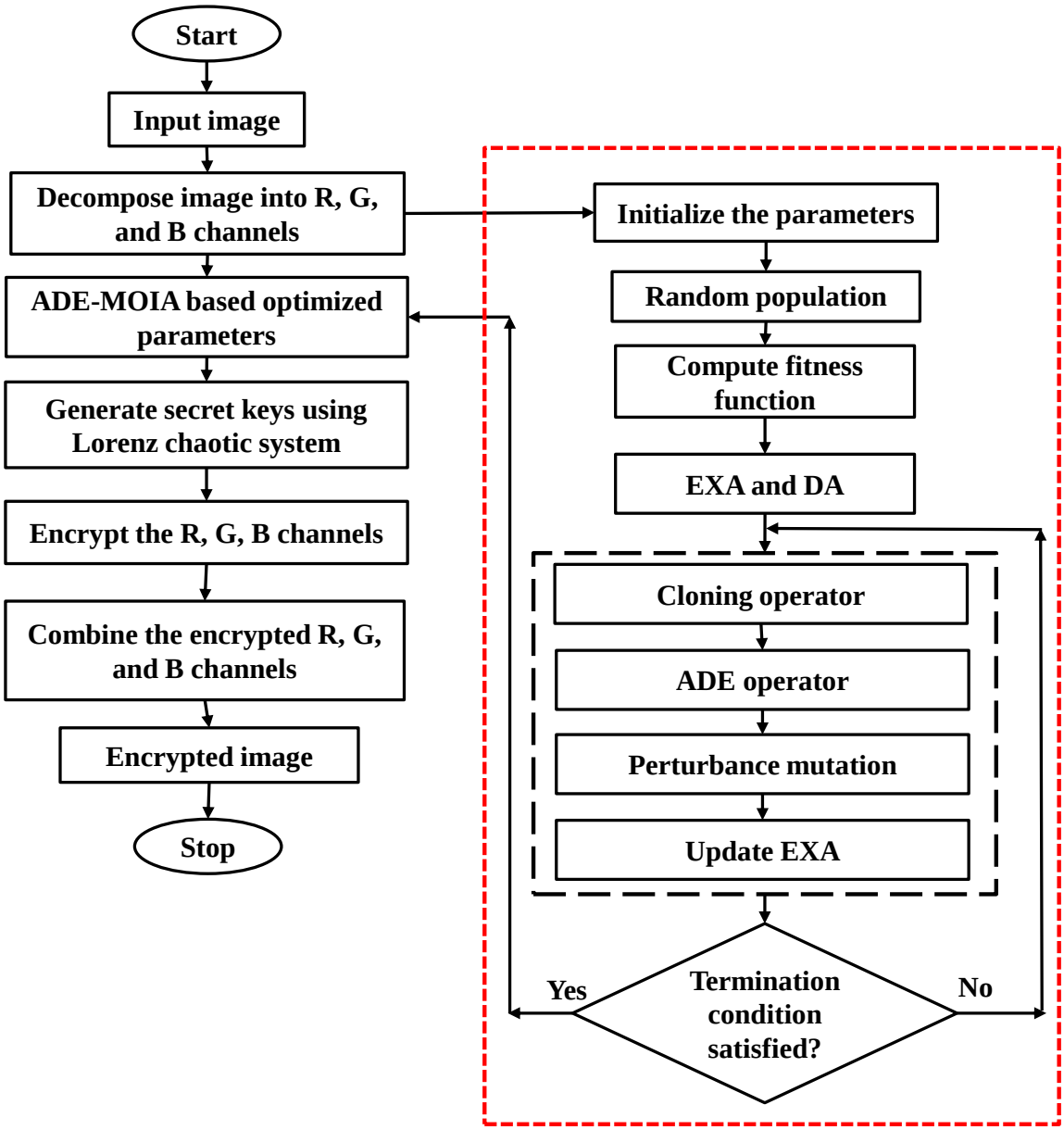



Figure 6.2: Flowchart of SHA-3, adaptive differential evolution, and Lorenz-like chaotic system based image encryption technique

**Algorithm 7:** Encryption algorithm

**Input:** Input image $I$

**Output:** Encrypted image $E$

//Decompose $I$ into red ($I_R$), green ($I_G$), and blue ($I_B$) channels.

$I_R = I(:, :, 1)$

$I_G = I(:, :, 2)$

$I_B = I(:, :, 3)$

generate optimal parameters ($a_{si}(i = 1 : N)$) by using **Algorithm** 8

generate secret keys $x$, $y$, and $z$ by assigning $a_{si}$ using **Algorithm 10**

//Use secret keys $x$, $y$, and $z$ to diffuse the pixel values of channels

$I_R$'= mod($I_R \oplus x$, 256)

$I_G$ = mod($I_G \oplus y$, 256)

$I_G$'= mod($I_G \oplus I_R$', 256)

$I_B$ = mod($I_B \oplus z$, 256)

$I_B$'= mod($I_B \oplus I_G$', 256)

//Update the secret keys $x$, $y$, and $z$

$x$' = $x \oplus y$

$y$' = $y \oplus z$

$z$' = $z \oplus x$

//Encrypt the $I_R$', $I_G$', and $I_B$' using updated secret keys $x$', $y$', and $z$'

$\alpha_e$ = $a_{sN}$

$E_R$ = mod($x$'× $I_R$' + $(1 - \alpha_e)$× $x$', 256)

$E_G$ = mod($y$'× $I_G$' + $(1 - \alpha_e)$× $y$', 256)

$E_B$ = mod($z$'× $I_B$' + $(1 - \alpha_e)$× $z$', 256)

//Concatenate the encrypted channels to obtain the encrypted image

$E$ = cat($E_R$, $E_G$, $E_B$)

return $E$

### 6.3.1 Key generation process using adaptive differential evolution with multi-objective immune algorithm

ADE-MOIA is used to optimize the parameters of Lorenz-like chaotic system (*i.e.,* Eq. 1.11) for better secret keys generation. Algorithm 8 describes the process of optimized key generation using ADE-MOIA algorithm. The initial values of External archives (EXA) and Dominated archives (DA) are generated using Algorithm 9.

ADE-MOIA is then utilized to optimize the solutions obtained. Thereafter, cloning operator is applied on external archives to generate cloning population using Algorithm 12.

**Algorithm 8:** Adaptive differential evolution with multi-objective immune algorithm based key generation

**Input:** Input image $I$

**Output:** Optimal parametrs ($EXA_m$)

Generate initial population, *i.e.,* $EXA$ and $DA$ using **Algorithm 9**
$gen$ = 0
$s_v$ = {}
**while** *gen* $> max_{gen}$ **do**
    set the value of $CR$ using Eq. (6.14)
    **for** *i = 1 to N* **do**
        apply cloning operator on EXA using **Algorithm 12**
        set $F_i$ using Eq. (6.15)
        generate $v_i$ (*i.e.,* ADE operator) using Eq. (6.12)
        generate $\Psi_i$ using Eq. (6.13)
        perform polynomial mutation on $\Psi_i$ to get $\rho_i$ using **Algorithm 13**
        evaluate $\rho_i$ using **Algorithm 11**
        add $\rho_i$ to the child population $P_{child}$.
        **if** $\rho_i \succ a_{si}$ **then**
            $s_v = s_v \cup F_i$
        **end**
    **end**
    **if** $|s_v|$>*p %.N* **then**
        update $F_m$ using Eq. (6.16)
        $s_v$ = {}
    **end**
    update EXA using **Algorithm 14**
    set $gen = gen$+1
**end**
Updated EXA
// EXA contains non-dominated solutions.
//Select middle solution of EXA as an optimal solution.
$EXA_m$ = EXA(size(EXA)/2)
return $EXA_m$

The polynomial mutation is applied on population to generate new solutions (see Algorithm 13). The values of Crossover rate ($CR$) and scaling factor ($F$) are obtained by using Eqs. (6.14) and (6.15). Afterward, the intermediate solution ($\Psi_i$) is evaluated using Eqs. (6.12) and (6.13). It is further disturbed to obtain child ($\rho_i$) (see Algorithm 13). The objectives of $\rho_i$ are obtained and stored in child population ($P_{child}$). In case, if a parent ($a_{si}$) is dominated by $\rho_i$, then the corresponding value of $F$ is saved in success value set ($s_v$). If

size($s_v$) $> p\%$ of $N$, then value of $F_m$ is evaluated using Eq. (6.16). An archive updation process is done through Algorithm 14. The whole process repeats until the termination criteria satisfied. Finally, the non-dominated solutions ($EXA$) are returned.

#### 6.3.1.1 Population initialization

An initial population $P = \{a_{s1}, a_{s2}, \ldots, a_{sN}\}$ is created in which the decision variables of $a_{si}(i = 1, 2, \ldots, N)$ are developed using random distribution. Then, the objective values of each $P_i$ are evaluated. The non-dominated and dominated solutions are obtained by using non-dominated sorting [178] in $P$. These solutions are stored in $EXA$ and $DA$, respectively.

**Algorithm 9:** Population initialization

**Input:** Solution size $N$

**Output:** EXA and DA

```
for i = 1 to N do
    randomly generate a solution a_si
    evaluate the objectives of a_si using Algorithm 11
    add a_si to the population P
end
EXA = Find_nonDominated(P)
DA = Find_Dominated(P)
```

#### 6.3.1.2 Key generation

SHA-3 is used to generate 256-bit external secret key from the input image. The main benefit of using SHA-3 is that if there is only one-bit difference between two plain images, then it will produce completely different hash values. The initial conditions of Lorenz-like chaotic system are generated through 256-bit external secret key. ISAL is highly sensitive towards the plain image. 256-bit secret key ($S_k$) is decomposed into 8-bit blocks ($s_{ki}$), which is given below:

$$S_k = s_{k1}, s_{k2}, \ldots, s_{k32} \tag{6.1}$$

where $s_{ki} = \{s_{ki,0}, s_{ki,1}, \ldots, s_{ki,7}\}$. The initial conditions of Lorenz-like system can be obtained as follows:

$$x_0 = x_0' + \frac{(s_{k1} \oplus s_{k2} \oplus \cdots \oplus s_{k11})}{256} \tag{6.2}$$

$$y_0 = y_0' + \frac{(s_{k12} \oplus s_{k13} \oplus \cdots \oplus s_{k22})}{256} \tag{6.3}$$

$$z_0 = z_0' + \frac{(s_{k23} \oplus s_{k24} \oplus \cdots \oplus s_{k32})}{256} \tag{6.4}$$

$$a = (x_0 + y_0) \ mod \ 256 \tag{6.5}$$

$$c = (x_0 + z_0) \ mod \ 256 \tag{6.6}$$

where $x'_0$, $y'_0$, and $z'_0$ are initial values that are randomly generated using Algorithm 2. The other parameters such as $l_1$, $l_2$, and $\delta$ required by Lorenz-like chaotic system are also obtained in similar way.

**Algorithm 10:** Key generation

**Input:** Solution $a_{si}(i = 1 : N - 1)$

**Output:** Secret keys $x$, $y$, and $z$

1 //In solution $a_{si}(i = 1 : N - 1)$, $N$ = 7.
2 //Here, $a_{s1}$, $a_{s2}$, and $a_{s3}$ represent $x'_0$, $y'_0$, and $z'_0$.
3 //$a_{s4}$, $a_{s5}$, and $a_{s6}$ represent $l_1$, $l_2$, and $\delta$.
4 $l_1 = a_{s4} * 100$
5 $l_2 = a_{s5} * 100$
6 Generate initial values $x_0$, $y_0$, $z_0$, $a$, and $c$ using Eqs. (6.1)-(6.6 )
7 Generate secret keys using Eq. (1.11)
8 return ($x$, $y$, $z$)

#### 6.3.1.3 Multi-objective fitness function

In this technique, $H(S)$, NPCR, UACI, and $r_{x,y}$ are used to evaluate the fitness of obtained solutions. The fitness function is defined as:

$$\begin{aligned} \text{Maximize } \ f(\varphi) &= \frac{H(S)}{8} + \left(\frac{NPCR + UACI}{2}\right) \\ \text{subject to: } \ -0.005 &\leq r_{x,y} \leq 0.005 \\ H(S) &> 7.9 \end{aligned} \tag{6.7}$$

**Algorithm 11:** Fitness evaluation

**Input:** Input image $I$ and solution $a_{si}(i = 1 : N)$

**Output:** Fitness value $f(\varphi)$

1 Generate an encrypted image using **Algorithm 7**
2 $e_n$ = entropy($E$)
3 $n_p$ = NPCR($E$)
4 $u_a$ = UACI($E$)
5 $cc = r_{x,y}(E)$
6 **if** *-0.005≤cc≤0.005 &&* $e_n$*>7.9* **then**
7 $\quad f(\varphi) = \frac{e_n}{8} + \frac{n_p + u_a}{2}$
8 **end**
9 return $f(\varphi)$

To normalize these values, $H(S)$ is divided by $8$ and the summation of NPCR and UACI is divided by $2$. There are two constraints. The first constraint is $H(S)$>$7.9$. The other constraint is correlation coefficient i.e., $r_{x,y}$ (see Eq. 1.15) which should be closed to $0$. Because the relation between adjacent pixels of an encrypted image should be random in nature.

#### 6.3.1.4 Cloning

Let us assume that population, $P$ with size $N$ and cloning population $P_C = \{a_1, a_2, \ldots, a_{NC}\}$ with size $NC$. At first, $NC$ antibodies having maximum value of crowding-distance ($CD$) are selected from $EXA$. The cloning operator is then used to evaluate $P$.

$$P = \cup_{i=1}^{NC}\{q_i \times q_i\} \tag{6.8}$$

where $q_i$ represents the number of clones with respect to each antibody $a_i(i = 1, 2, \ldots, NC)$. $q_i$ is defined as:

$$q_i = \left\lceil N \times \frac{CD(a_i)}{\sum_{j=i}^{NC} CD(a_j)} \right\rceil \tag{6.9}$$

where $CD(a_i)$ represents the fitness values of $a_i$. $CD$ is computed as:

$$CD(a_i) = \sum_{j=1}^{m} \frac{CD_j(a_i)}{f_j max - f_j min} \tag{6.10}$$

where $m$ represents the number of objectives. $f_{jmax}$ and $f_{jmin}$ are maximal and minimal values of $j^{th}$ objective, respectively. $CD_j(a_i)$ represents the $CD$ of $j^{th}$ objective for $a_i$ and is evaluated as:

$$CD_j(a_i) = \begin{cases} \infty, \quad \text{if}(f_j(a_i) == f_j min \text{ or } f_j(a_i) == f_j max); \\ min\{f_j(a_k) - f_j(a_l)\}, f_j(a_k) > f_j(a_i) > f_j(a_l), \text{otherwise} \end{cases} \tag{6.11}$$

Here, $k, l \in [1, N]$. If $a_i$ is located within the boundary, then the corresponding $CD$ approaches towards $\infty$. Due to this, the number of clones are not obtainable with the help of Eq. (6.11). Therefore, $a_i$ is set as twice of the maximum value of $CD$ except for the boundary solutions. Algorithm 12 describes the cloning operator.

**Algorithm 12:** Cloning operator

**Input:** External archives EXA

**Output:** Cloning population $P$

$P_C = EXA$
**if** $|P_C|$>*NC* **then**
  CrowdingDistanceAssignment $(P_C)$
  //Sort $P_C$ using $CD$ in descending order
  $P_C$ = Sort($P_C$)
  //Select the first $NC$ antibodies in $P_C$. $P_C$ = SelectforClone($P_C$)
**end**
**for** *i = 1 to* $|P_C|$ **do**
  Calculate the number $q_i$ of clones for $a_{si}$
  Clone $q_i$ solutions of $a_{si}$ and add them to $P$
**end**

#### 6.3.1.5 Evolutionary operator

ADE-MOIA adaptively controls the evolutionary parameters such as CR and F of differential evolution. It also uses the adaptive differential operator to select the parents so that better-optimized solution can be obtained.

i. Differential evolution operator

In ADE-MOIA, an adaptive differential evolution (ADE) operator is used to improve the parent selection strategy. ADE operator is computed as:

$$v_i = a_{s_{r_1}} + F(a_{s_{r_2}} - a_{s_{r_3}}) \tag{6.12}$$

where $F \in (0, 1.0)$. The selected parent vectors have significant impact on $EXA$. To improve the diversity of $P$, three parent vectors (*i.e.,* $a_{s_{r_1}}$, $a_{s_{r_2}}$, and $a_{s_{r_3}}$) are selected from $EXA$ and $DA$. Also, $a_{s_{r_1}}$ and $a_{s_{r_2}}$ are selected from $EXA$. $a_{s_{r_3}}$ is obtained from $DA$. Thereafter, a trial point ($\Psi_i$) is evaluated by using its parents $a_{si}$ and $v_i$, which is computed as:

$$\Psi_i^j = \begin{cases} v_i^j, & \text{if}(R_j < CR \ \ or \ \ j = I_i) \\ a_{si}^j, & \text{otherwise.} \end{cases} \tag{6.13}$$

where $j$ represents the $j^{th}$ variable of decision vectors. $I_i \in [1, n]$ guarantees at least one dimension of trial vector is calculated from the mutant vector. $R_j \in (0, 1.0)$ for $j^{th}$ variable. The crossover rate ($CR$) lies in the range of $[0, 1.0]$.

ii. Adaptive parameters control

To control the value of $CR$, it is always re-evaluated during the evolution process. For the child solution, $CR$ decides how much information will be obtained from its respective

parents. $CR$ is computed as:

$$CR_{gen} = 0.55 \times \frac{1}{\pi} \times arctan\left(\frac{1 - gen/maxgen - 0.7}{0.1}\right) \tag{6.14}$$

where $gen$ and $maxgen$ represent current and maximum number of iterations, respectively. The value of $CR$ in each iteration is significantly reduced with the increase in number of iterations. Thus, larger $CR$ states that the children are evaluated more randomly. Therefore, $ADE$ encourages global search. By the half of $maxgen$, children will inherit more information from their respective parents. The local search is achieved at the end of evolution process.

$F$ is adaptively evaluated as:

$$F_i = Cauchy(F_m, 0.1), \;\; F_i \in (0.1, 0.9) \tag{6.15}$$

Initially, the value of $F_m$ is set to $0.5$. During the course of iterations, $F_m$ is updated using the improvement situation of children which is given below:

$$F_m = average(s_v) = \sum_{x \in s_v} (x/|s_v|) \tag{6.16}$$

The value of $F$ is stored into $s_v$ only when the generated child by current value of $F$ is a non-dominated to its respective parent. If size $(s_v) > p\%$ of the population, then $F_m$ is re-evaluated by using Eq. (6.16) and each $F_i$ is evaluated using the updated value of $F_m$. In ISAL, the value of $p\%$ is set to $10\%$ based on trial and error.

#### 6.3.1.6 Polynomial mutation

The polynomial mutation ($P_M$) is used to develop new solutions and is obtained by:

$$\rho_k = \begin{cases} \Psi_k + \sigma_k \times (ub_k - lb_k), & \text{if } \; r < p_m \\ \Psi_k, & \text{otherwise.} \end{cases} \tag{6.17}$$

where $\rho_k$ and $\Psi_k$ represent $k^{th}$ decision variables before and after mutation, respectively. $ub_k$ and $lb_k$ represent upper and lower bounds, respectively . $r$ is a random variable whose lies in the range of $[0, 1.0]$. $\sigma_k$ is a small variation that is computed as:

$$\sigma_k = \begin{cases} (2 \times r)^{\frac{1}{\eta}-1}, & \text{if } \; r < 0.5 \\ 1(2 - 2 \times r)^{\frac{1}{\eta}}, & \text{otherwise.} \end{cases} \tag{6.18}$$

Here, mutation parameter ($\eta$) controls the magnitude of $P_M$. The maximum value of $\eta$ depicts minimum variance. The polynomial mutation ($P_M$) is described in Algorithm 13. $n$ and $p_m$ represent dimension of variables and probability for $P_M$, respectively.

**Algorithm 13:** Polynomial mutation

**Input:** Decision variable before mutation $\Psi_k$
**Output:** Decision variable after mutation $\rho_k$

```
for k=1 to n do
    if random(0,1)< p_m then
        calculate σ_k using Eq. (6.18)
        calculate ρ_k using Eq. (6.17)
        if ρ_k > ub_k then
            ρ_k = ub_k
            else if ρ_k<lb_k then
                ρ_k = lb_k
            end
        end
    end
end
```

#### 6.3.1.7 Archive update

After $ADE$ and $P_M$ operators, $o_s$ and $a_i$ in $EXA$ are integrated. Only one solution is selected in case of identical solutions. The fast non-dominated sorting technique is used to store these non-dominated solutions in $EXA$ and other evaluated dominated solutions. If size of $EXA$ is greater than $N$, then a fine-gained selection [179] is used to remain the population diversity.

It is achieved by iteratively removing the most crowded solution and also by updating its neighbor's $CD$ until the termination criterion (*i.e.,* $size(EXA) \leq N$) is satisfied. At the end of iteration, only the solutions with maximum $CD$s are retained in $EXA$. The archive updating process is represented in Algorithm 14.

**Algorithm 14:** Archive updating process

**Input:** EXA and $P_{child}$
**Output:** Updated EXA

```
//Combine the offspring and EXA to A
A = Union(EXA,P_child)
//Find nondominated antibodies in A
EXA = Find_nonDominated(A)
//Find dominated solution in A
DA = Find_Dominated(A)
while (|EXA| > N) do
    CrowdingDistanceAssignment(EXA)
    EXA = Sort(EXA)
    EXA = DeleteCrowded(EXA)
end
```

### 6.3.2 Image decryption algorithm

To decrypt an encrypted image at receiver side, the same secret keys are required which used in encryption process. Therefore, the initial values, *i.e.,* $x'_0$, $y'_0$, $z'_0$, $l_1$, $l_2$, and $\alpha_e$ are sent along with an encrypted image to receiver. The decryption process is same as encryption process but in opposite direction. The image decryption approach is described in Algorithm 15.

**Algorithm 15:** Decryption approach

**Input:** Inital values $x'_0$, $y'_0$, $z'_0$, $l_1$, $l_2$, $\delta$, $\alpha_e$, and encrypted image ($E$)

**Output:** Decrypted image $D$

// Decompose $E$ into red $E_R$, green $E_G$, and blue channels $E_B$

$E_R = E(:, :, 1)$

$E_G = E(:, :, 2)$

$E_B = E(:, :, 3)$

generate secret keys $x$, $y$, and $z$ using **Algorithm 10**

// Generate other three secret keys $x'$, $y'$, and $z'$ from $x$, $y$, and $z$

$x' = x \oplus y$

$y' = y \oplus z$

$z' = z \oplus x$

//Decrypt $E_R$, $E_G$, and $E_B$ using $x'$, $y'$, and $z'$

$D_R = (E_R - (1-\alpha_e) \times x')/\alpha_e$

$D_G = (E_G - (1-\alpha_e) \times y')/\alpha_e$

$D_B = (E_B - (1-\alpha_e) \times z')/\alpha_e$

//Decrypt $D_R$, $D_G$, and $D_B$ using $x$, $y$, and $z$

$D'_R = D_R \oplus x$

$D''_G = D_R \oplus D_G$

$D'_G = D''_G \oplus y$

$D''_B = D_B \oplus D_G$

$D'_B = D''_B \oplus z$

//Combine decrypted channels to obtain the final decrypted image

$D = \text{cat}(D'_R, D'_G, D'_B)$

return $D$

## 6.4 Experimental results and discussion

In this section, experiments are performed to evaluate the performance of ISAL. ISAL has been tested on four color benchmark test images. The images are Sparrow, Tiger, Dog, and Flowers. The size of these images are $256 \times 256$.

The performance of ISAL is compared with five well-known meta-heuristic based image encryption techniques such as GA [47], ACO [121], WDICA [122], GDNA [46], and DHS

[124]. Figure 6.3 shows the encryption results obtained on color images using ISAL. Note that only red channel results of color images are considered in this work.


Figure 6.3: Plain images (a) Sparrow, (b) Tiger, (c) Dog, and (d) Flowers. Encrypted images (e) Sparrow, (f) Tiger, (g) Dog, and (h) Flowers

### 6.4.1 Performance evolution

The performance of ISAL is evaluated using entropy, peak signal to noise ratio, and mean absolute error. The performance parameters are discussed in the following subsequent sub-sections.

#### 6.4.1.1 Entropy

Table 6.2 shows the entropy obtained from ISAL and the above-mentioned techniques. It can be observed from table that the entropy obtained from ISAL is near to ideal value. It implies that each pixel of an encrypted image carries the same amount of information. The attacker cannot extract the information using statistical attacks.

Table 6.2: Comparative analysis of ISAL with respect to entropy

| Technique | Sparrow | Tiger | Dog | Flowers |
|---|---|---|---|---|
| GA | 7.9970 | 7.9969 | 7.9974 | 7.9973 |
| ACO | 7.9834 | 7.9840 | 7.9844 | 7.9875 |
| WDICA | 7.9752 | 7.9739 | 7.9754 | 7.9759 |
| GDNA | 7.9693 | 7.9753 | 7.9788 | 7.9749 |
| DHS | 7.9971 | 7.9970 | 7.9968 | 7.9976 |
| ISAL | 7.9984 | 7.9981 | 7.9983 | 7.9985 |

#### 6.4.1.2 Peak signal to noise ratio

Table 6.3 shows the comparison of ISAL with the existing techniques in terms of PSNR. It is observed that ISAL provides better decrypted images as compared to other techniques. Table 6.3 indicates that DHS provides efficient decrypted images as compared to the existing techniques. However, ISAL shows significant improvement over DHS. The mean improvement of ISAL over DHS in terms of PSNR is found to be $13.9718$. Therefore, decrypted images obtained using ISAL contain little amount of noise as compared to competitive image encryption techniques.

Table 6.3: Comparative analysis of ISAL in terms of PSNR

| Technique | Sparrow | Tiger | Dog | Flowers |
|---|---|---|---|---|
| GA | $66.4563$ | $69.2006$ | $68.8924$ | $70.5974$ |
| ACO | $65.3574$ | $67.5280$ | $63.7852$ | $69.4719$ |
| WDICA | $70.2507$ | $71.4258$ | $67.3715$ | $69.9857$ |
| GDNA | $68.8452$ | $67.9259$ | $66.6521$ | $71.5822$ |
| DHS | $70.9041$ | $71.4073$ | $72.5417$ | $70.2150$ |
| ISAL | $84.1473$ | $86.7839$ | $87.9446$ | $85.8255$ |

#### 6.4.1.3 Mean absolute error

The encrypted image should be different from the input image. Mean Absolute Error (MAE) [63] is used to measure the difference between an input and encrypted images. The maximum value of MAE indicates that there is a significant difference between input and encrypted images. Table 6.4 shows the comparison between ISAL and other existing techniques. It can be observed from the table that ISAL has maximum MAE as compared to the existing image encryption techniques. The mean improvement of ISAL over existing image encryption techniques is found to be $4.787$.

Table 6.4: Comparative analysis of ISAL using MAE

| Technique | Sparrow | Tiger | Dog | Flowers |
|---|---|---|---|---|
| GA | $82.1455$ | $85.2533$ | $83.7810$ | $80.6587$ |
| ACO | $84.7425$ | $86.6588$ | $85.1148$ | $86.9980$ |
| WDICA | $83.2705$ | $81.8542$ | $84.1573$ | $85.7589$ |
| GDNA | $82.2548$ | $83.2589$ | $86.1456$ | $85.2285$ |
| DHS | $85.4091$ | $86.7043$ | $85.4571$ | $84.5012$ |
| ISAL | $89.9989$ | $92.7588$ | $94.9776$ | $95.8533$ |

### 6.4.2 Security analysis

The security analysis of ISAL is performed in this section to check the effectiveness of proposed technique against the various attacks.

#### 6.4.2.1 Histogram analysis

Figures 6.4 (b) and (d) show the histograms of plain and encrypted Sparrow images, respectively. Figures 6.5 (b) and (d) show the histograms of plain and encrypted Tiger images, respectively.

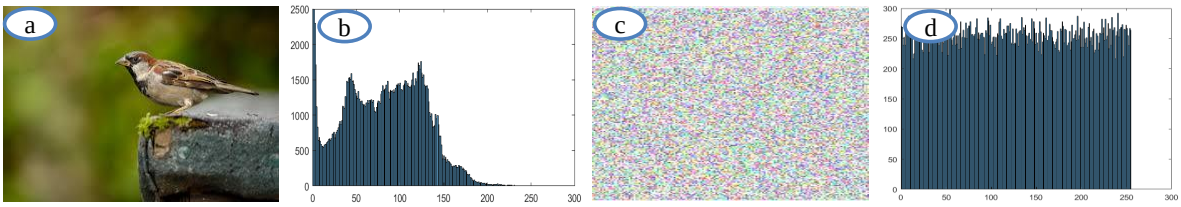



Figure 6.4: Sparrow image (a) Plain image, (b) Histogram of plain image, (c) Encrypted image, and (d) Histogram of encrypted image

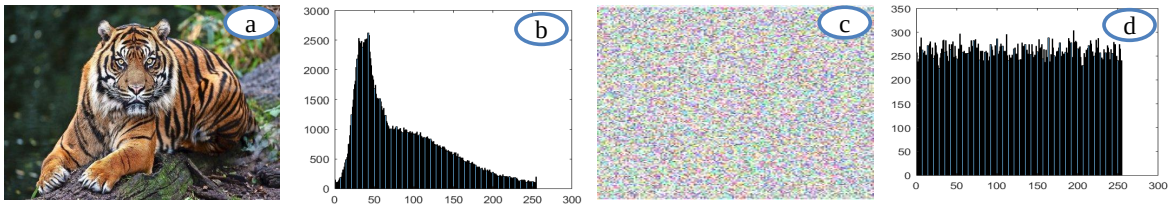



Figure 6.5: Tiger image (a) Plain image, (b) Histogram of plain image, (c) Encrypted image, and (d) Histogram of encrypted image

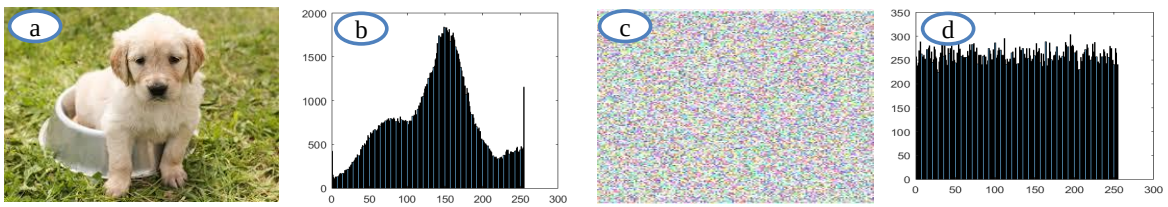



Figure 6.6: Dog image (a) Plain image, (b) Histogram of plain image, (c) Encrypted image, and (d) Histogram of encrypted image

Figures 6.6 (b) and (d) show the histograms of plain and encrypted Dog images, respectively. Figures 6.7 (b) and (d) show the histograms of plain and encrypted Flowers images, respectively. From Figures 6.4 (d)-6.7 (d), it can be observed that the encrypted images have uniform distribution of pixels. Therefore, the attacker cannot unveil any statistical information from the encrypted image.

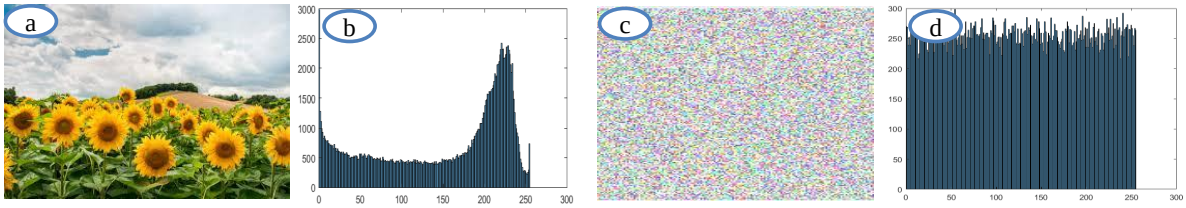



Figure 6.7: Flower image (a) Plain image, (b) Histogram of plain image, (c) Encrypted image, and (d) Histogram of encrypted image

#### 6.4.2.2 Correlation analysis

Table 6.5 shows the comparative analysis based on horizontal correlation. The horizontal correlation coefficient values are approximately equal to $0$.

Table 6.5: Comparative analysis of ISAL in terms of horizontal correlation

| Technique | Sparrow | Tiger | Dog | Flowers |
|---|---|---|---|---|
| GA | $0.0030$ | $0.0016$ | $0.0021$ | $-0.0005$ |
| ACO | $0.0135$ | $0.0054$ | $0.0136$ | $0.0019$ |
| WDICA | $0.0049$ | $0.0209$ | $0.0116$ | $-0.0127$ |
| GDNA | $0.0112$ | $0.0126$ | $0.0153$ | $-0.0110$ |
| DHS | $0.0104$ | $0.0215$ | $0.0042$ | $-0.0106$ |
| ISAL | $-0.0003$ | $0.0010$ | $0.0030$ | $0.0012$ |

Table 6.6: Comparative analysis of ISAL in terms of vertical correlation

| Technique | Sparrow | Tiger | Dog | Flowers |
|---|---|---|---|---|
| GA | $0.0041$ | $0.0027$ | $0.0032$ | $0.0008$ |
| ACO | $0.0110$ | $0.0047$ | $0.0039$ | $0.0021$ |
| WDICA | $0.0038$ | $0.0009$ | $0.0076$ | $0.0007$ |
| GDNA | $0.0107$ | $0.0131$ | $0.0103$ | $0.0100$ |
| DHS | $0.0029$ | $0.0015$ | $0.0013$ | $0.0002$ |
| ISAL | $0.0026$ | $-0.0033$ | $0.0006$ | $0.0001$ |

Tables 6.6 and 6.7 show the performance comparison of ISAL with the other existing techniques based on vertical and diagonal correlation, respectively. It is observed from these tables that the correlation of ISAL is near to zero or negative value. Hence, there is no statistical information can be recovered from the encrypted image.

Table 6.7: Comparative analysis of ISAL in terms of diagonal correlation

| Technique | Sparrow | Tiger | Dog | Flowers |
|---|---|---|---|---|
| GA | 0.0058 | 0.0064 | 0.0039 | 0.0050 |
| ACO | 0.0120 | 0.0045 | 0.0152 | 0.0214 |
| WDICA | 0.0075 | 0.0216 | 0.0124 | 0.0116 |
| GDNA | 0.0133 | 0.0147 | 0.0131 | 0.0050 |
| DHS | 0.0120 | 0.0117 | 0.0029 | 0.0026 |
| ISAL | 0.0043 | 0.0038 | 0.0013 | 0.0022 |

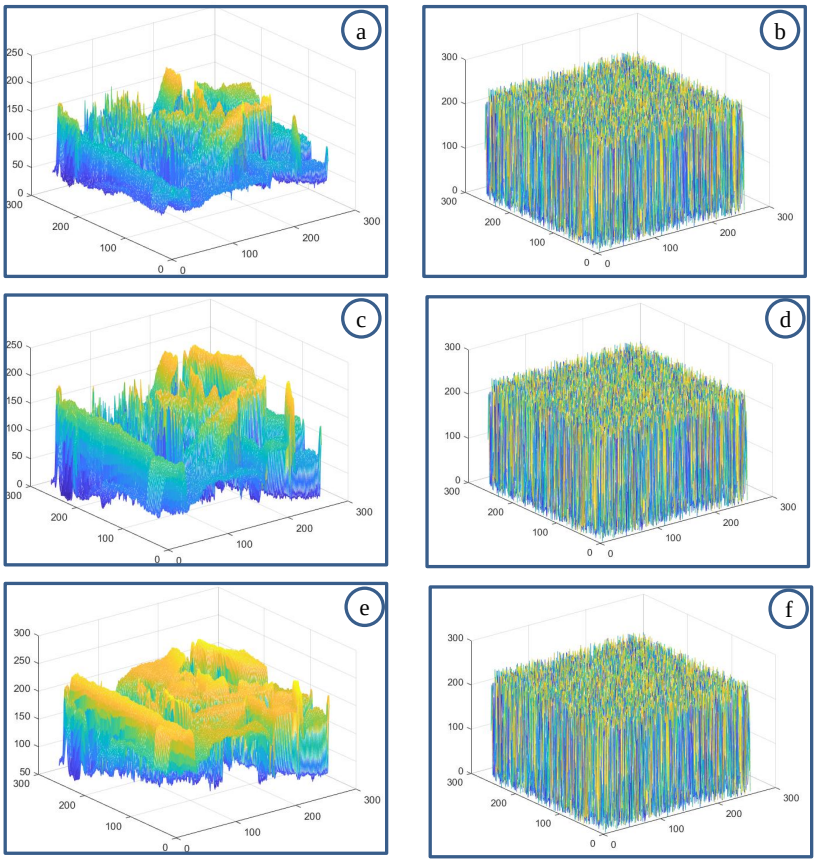



Figure 6.8: Correlation analysis of plain and encrypted Tiger image (a) plain horizontal correlation, (b) Plain diagonal correlation, (c) Plain vertical correlation, (d) Encrypted horizontal correlation, (e) Encrypted diagonal correlation, and (f) Encrypted vertical correlation

Figures 6.8 (a), (c), and (e) show the horizontal, diagonal, and vertical correlation of plain Tiger image. From figures, it can be seen that the adjacent pixels of an original image are highly related to each other. Figures 6.8 (b), (d), and (e) show the horizontal, diagonal, and vertical correlation analysis of encrypted Tiger image. From here, it can be observed that the adjacent pixels of encrypted images obtained from ISAL are loosely related to each other. Hence, ISAL does not leak any kind of statistical information to an attacker.

#### 6.4.2.3 Differential analysis

Tables 6.8 and 6.9 show the comparison of ISAL with other encryption techniques on the basis of NPCR and UACI, respectively. From Table 6.8, it has been observed that the mean improvement of ISAL in terms of NPCR over GA, ACO, WDICA, GDNA, and DHS is 0.0013, 0.0019, 0.0012, 0.0034, and 0.0009, respectively. From Table 6.9, the mean improvement is observed in ISAL with respect to UACI over GA, ACO, WDICA, GDNA, and DHS is 0.156%, 0.0045, 0.0024, 0.0037, 0.0017, and 0.0008, respectively. It can be observed from these tables that if the attacker tries to change only one bit in the input image, then ISAL will generate completely different image. ISAL is extremely sensitive towards the small change in the plain image.

Table 6.8: Comparative analysis of ISAL using NPCR

| Technique | Sparrow | Tiger | Dog | Flowers |
|---|---|---|---|---|
| GA | 0.9945 | 0.9953 | 0.9940 | 0.9952 |
| ACO | 0.9930 | 0.9939 | 0.9950 | 0.9949 |
| WDICA | 0.9956 | 0.9954 | 0.9953 | 0.9950 |
| GDNA | 0.9934 | 0.9943 | 0.9931 | 0.9930 |
| DHS | 0.9960 | 0.9962 | 0.9961 | 0.9959 |
| ISAL | 0.9968 | 0.9970 | 0.9972 | 0.9965 |

Table 6.9: Comparative analysis of ISAL using UACI

| Technique | Sparrow | Tiger | Dog | Flowers |
|---|---|---|---|---|
| GA | 0.3311 | 0.3320 | 0.3325 | 0.3312 |
| ACO | 0.3332 | 0.3334 | 0.3329 | 0.3326 |
| WDICA | 0.3319 | 0.3322 | 0.3315 | 0.3328 |
| GDNA | 0.3340 | 0.3342 | 0.3348 | 0.3339 |
| DHS | 0.3349 | 0.3352 | 0.3343 | 0.3351 |
| ISAL | 0.3356 | 0.3360 | 0.3357 | 0.3355 |

#### 6.4.2.4 Secret key space

In ISAL, six secret keys such as $x$, $y$, $z$, $x'$, $y'$, and $z'$ are used to encrypt the color images. The size of secret keys is same as the input image, *i.e.,* $256 \times 256$. Therefore, the key space is given below

$$\text{Key space} = (256! \times 256!)^6 \approx (2^{3369})^6 \tag{6.19}$$

Thus, ISAL has a large key space to resist against brute-force attacks.

#### 6.4.2.5 Secret key sensitivity

The sensitivity of ISAL towards secret keys has been evaluated. The initial parameters such as $x'_0$, $y'_0$, $z'_0$, $l_1$, $l_2$, and $\delta$ are required to generate three secret keys, *i.e.,* $x$, $y$, and $z$ for Lorenz-like chaotic system. $I_R$, $I_G$, and $I_B$ channels of $I$ are encrypted using $x$, $y$, and $z$, respectively. The encrypted image $E$ is obtained from the combination of encrypted channels. Figure 6.9 (b) shows the encrypted image using original secret keys.

If the slight change is made in the initial parameter, *i.e.,* $x'_0$, then three different secret keys such as $x'$, $y'$, and $z'$ are generated. Using these modified secret keys, $I_R$, $I_G$, and $I_B$ channels of same input image are encrypted. The final encrypted image $E'$ is obtained from the combination of encrypted channels.

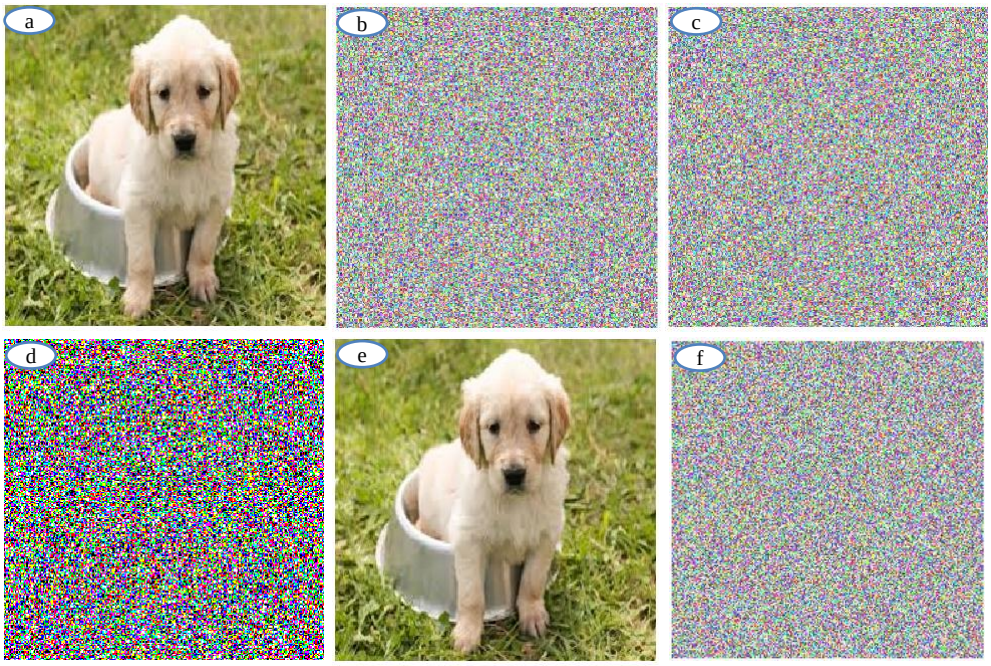



Figure 6.9: Secret key sensitivity analysis of ISAL (a) Dog plain image, (b) Encrypted image with secret key, (c) Encrypted image using modified secret key with the difference of one pixel, (d) Difference between (b) and (c), (e) Decrypted image with original key, and (f) Decrypted image with modified key

Figure 6.9 (c) shows the encrypted image using modified secret keys. Figure 6.9 (d) shows the difference between two encrypted images, *i.e.,* $E$ and $E'$ which has small difference in the initial values. Figure 6.9 (e) shows the decrypted image of Figure 6.9 (b) using original secret keys. Figure 6.9 (f) shows the decrypted image which is obtained by applying the modified secret keys on original encrypted image (Figure 6.9). It can be observed that if there is small change in the initial values of secret keys, the encrypted image cannot be recovered.

Table 6.10 shows the quantitative difference between two encrypted images, *i.e.,* $E$ and $E'$. From the table, it can be seen that the difference between two encrypted images is almost $100$ %. Hence, ISAL is highly sensitive towards initial values.

Table 6.10: Difference between encrypted images using secret keys with small change

| | Sparrow | Tiger | Dog | Flowers |
|---|---|---|---|---|
| Difference | 99.9954 | 99.9968 | 99.9975 | 99.9963 |

#### 6.4.2.6 Noise and enhancement attacks analysis

In real applications, the images are inevitably attacked by various attacks such as Gaussian noise, compression, image enhancement, cropping, *etc.* An image encryption said to be robust if the input image can still be decrypted when the encrypted image is attacked. In subsequent sections, ISAL has been tested against various attacks.

**A. Noise attack analysis**

The robustness of ISAL is tested by decrypting the encrypted image contaminated by Salt and pepper, and Gaussian noise.

*i. Salt and pepper noise analysis:*

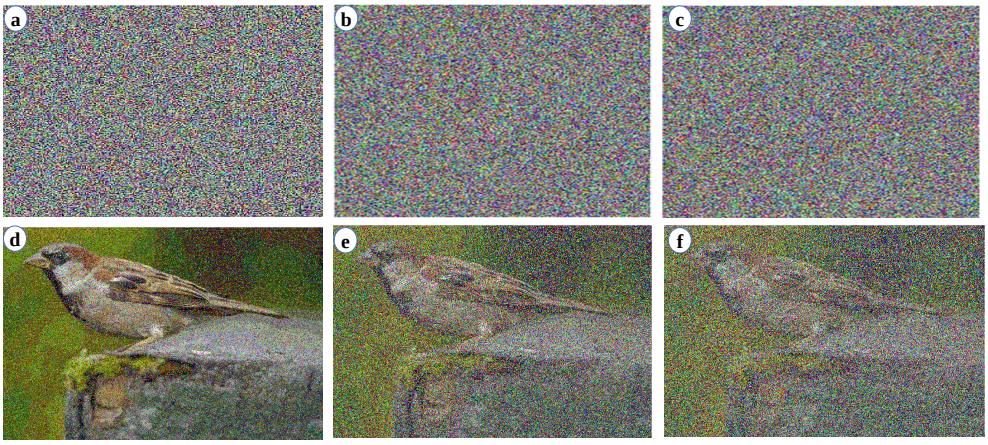



Figure 6.10: Salt and pepper noise analysis of ISAL (a) Attacked encrypted image (with density = 0.3), (b) Attacked encrypted image (with density =0.5), (c) Attacked encrypted image (with density = 0.7), (d) Attacked decrypted image obtained from (a), (e) Attacked decrypted image obtained from (b), and (f) Attacked decrypted image obtained from (c).

Figure 6.10 shows encrypted images contaminated by Salt and pepper noise with different noise densities such as $0.3$, $0.5$, and $0.7$, respectively. ISAL is employed to decrypt these noisy encrypted images (*i.e.,* Figures 6.10 (a)-(c)). The decrypted images are shown in Figures 6.10 (d)-(f), respectively. The corresponding PSNR values between the decrypted and input sparrow images are $29.36$ dB, $19.37$ dB, and $9.86$ dB, respectively. It clearly shows that the quality of decrypted image is degraded with the increase in noise density. However, the decrypted is still recognizable upto $70\%$ noise affected encrypted image.

*ii. Gaussian noise analysis*: Figure 6.11 shows encrypted images contaminated by Gaussian noise with different noise densities, *i.e.,* mean ($\mu$) = $0.1$ and variance ($\sigma^2$) = $0.1$, $\mu$ = $0.5$ and $\sigma^2$ = 0.1, and $\mu$ = $0.5$ and $\sigma^2$ = $0.5$. ISAL is employed to decrypt these noisy encrypted images (*i.e.,* Figures 6.11 (a)-(c)). The decrypted images are shown in Figures 6.11 (d)-(f), respectively. The corresponding PSNR values between the decrypted and input sparrow images are $27.81$ dB, $21.49$ dB, and $10.17$ dB, respectively. It clearly shows that the quality of decrypted image is degraded with the increase of mean and variance. However, the decrypted is still recognizable at higher value of $\mu$ and $\sigma^2$, respectively.

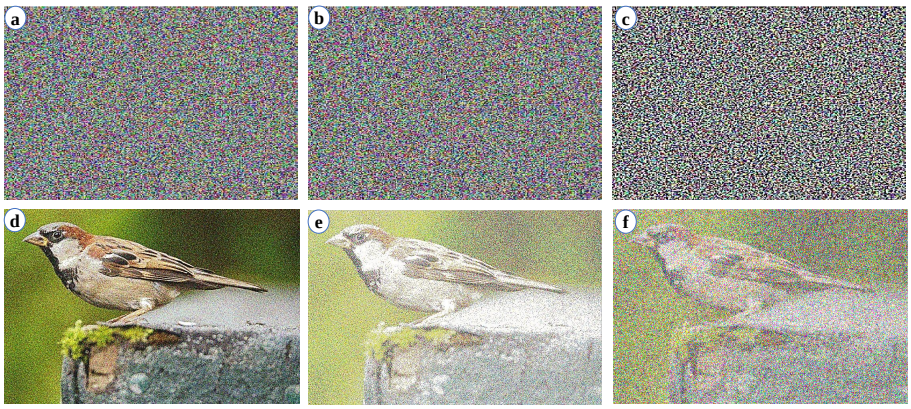



Figure 6.11: Guassian noise attack analysis of ISAL (a) Attacked encrypted image (with $\mu$ = 0.1 and $\sigma^2$ = 0.1), (b) Attacked encrypted image (with $\mu$ = 0.5 and $\sigma^2$ = 0.1), (c) Attacked encrypted image (with $\mu$ = 0.5 and $\sigma^2$ = 0.5), (d) Attacked decrypted image obtained from (a), (e) Attacked decrypted image obtained from (b), and (f) Attacked decrypted image obtained from (c)

**B. Enhancement attack analysis**

The image enhancement techniques improves the visibility of images. However, image enhancement techniques are widely used by attackers to degraded the quality of encrypted images. In this section, the robustness of ISAL is tested by decrypting the encrypted images obtained from image enhancement techniques.

Figure 6.12 shows encrypted images which are modified by applying the enhancement techniques such as histogram equalization, gamma correction and adaptive histogram equalization, respectively. ISAL is employed to decrypt these enhanced attack effected encrypted images (*i.e.,* Figures 6.12 (a)-(c)). The decrypted images are shown in Figures

6.12 (d)-(f), respectively. The corresponding PSNR values between the decrypted and input sparrow images are $23.49$ dB, $18.27$ dB, and $16.19$ dB, respectively. It is observed that the effect of adaptive histogram equalization is quite more than histogram equalization and gamma correction attacks.

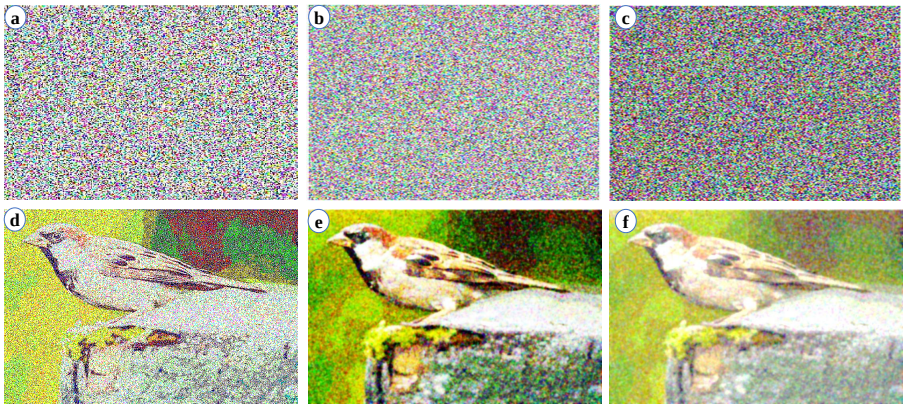



Figure 6.12: Enhancement attack analysis of ISAL (a) - (c) Attacked encrypted images obtained using (a) Histogram equalization, (b) Gamma correction, (c) Adaptive histogram equalization, (d)-(f) Attacked decrypted image (d) Attacked decrypted image obtained from (a), (e) Attacked decrypted image obtained from (b), and (f) Attacked decrypted image obtained from (c)

#### 6.4.2.7 Execution time

Execution time ($ET$) is measured as the time (in seconds) taken to execute a given image encryption technique. It is the aggregation of compile ($CT$) and runtime ($RT$) of the given technique. The 'tic' and 'toc' operators in MATLAB script have been used to evaluate $ET$.

Table 6.11: Encryption execution time analysis of ISAL

| Image | Size | GA | ACO | WDICA | GDNA | DHS | ISAL |
|---|---|---|---|---|---|---|---|
| Sparrow | $256 \times 256$ | 22.8 | 25.4 | 21.7 | 23.6 | 25.7 | 18.2 |
| Tiger | $512 \times 512$ | 58.9 | 60.9 | 56.9 | 55.7 | 58.6 | 40.4 |
| Dog | $1024 \times 1024$ | 648.6 | 702.7 | 637.8 | 639.9 | 625.9 | 613.6 |
| Flowers | $2048 \times 2048$ | 3669.8 | 3858.8 | 3559.4 | 3762.3 | 3657.3 | 3427.2 |

Tables 6.11 and 6.12 show the analysis of $ET$ in seconds for encryption and decryption process, respectively. It is observed that ISAL takes lesser time as compared to others. The mean reduction in $ET$ by using ISAL over the existing techniques is $1.3487$. Due to iterative process of meta-heuristic based image encryption techniques, these techniques take more time for large size images (see Table 6.11). However, image size do not affect the execution time of decryption process. It is observed from Tables 6.11 and 6.12 that ISAL is computationally faster than others.

Table 6.12: Comparative analysis of ISAL with respect to decryption time

| Image | Size | GA | ACO | WDICA | GDNA | DHS | ISAL |
|---|---|---|---|---|---|---|---|
| Sparrow | $256 \times 256$ | 0.058 | 0.049 | 0.039 | 0.043 | 0.038 | 0.029 |
| Tiger | $512 \times 512$ | 0.052 | 0.048 | 0.039 | 0.049 | 0.039 | 0.029 |
| Dog | $1024 \times 1024$ | 0.054 | 0.046 | 0.039 | 0.049 | 0.036 | 0.032 |
| Flowers | $2048 \times 2048$ | 0.057 | 0.045 | 0.035 | 0.047 | 0.041 | 0.033 |

### 6.4.3 Parallel analysis of ISAL

To simulate ISAL in parallel environment, extensive experiments have been carried out on Intel(R) Xeon CPU ES-2630 V4 @2.20GHZ with $10$ cores and GPU Nvidia graphics card 8 GB. Message passing interface has been used to run ISAL in parallel fashion. Parameter initialization, Arnold transform, conquer results, and termination criterion operations are run on a single core (*i.e.,* master). The master core decomposes the population in $9$ sub populations and assign them to slave cores. Each slave core is responsible for applying different adaptive differential operations such as cloning operator, ADE operator, perturbance mutation and update EXA in concurrent fashion. Table 6.13 shows the comparison of execution time between sequential and parallel ISAL in seconds (sec.). It is observed that parallel version of ISAL significantly improves the computational speed as compared to the sequential version of ISAL. Sparrow image is considered for parallel analysis. It is observed that the parallel ISAL has significant improvement in computational speed as the size of image increases.

Table 6.13: Comparison between sequential and parallel ISAL

| Image size | Sequential(in sec.) | Parallel(in sec.) |
|---|---|---|
| $256 \times 256$ | 16 | 11 |
| $512 \times 512$ | 40 | 32 |
| $1024 \times 1024$ | 613 | 193 |
| $2048 \times 2048$ | 3427 | 2037 |
| $4096 \times 4096$ | 18793 | 12031 |

## 6.5 Summary

In this chapter, Lorenz-like chaotic system based on ADE has been used to encrypt the images. The main benefits of ISAL are sensitive towards the input image and secret keys. It adaptively selects input parameters of Lorenz-like chaotic system. The different types of security analyses such as statistical, differential attack, noise attack, and image enhancement, have been performed on ISAL. The comparative results reveal that ISAL provides

better encryption results than the others. ISAL has the ability to resist against statistical, differential, noise, and image enhancement attacks.

# Chapter 7

# Non-dominated sorting genetic algorithm based image encryption

## 7.1 Introduction

From literature, it has been observed that the majority of existing image encryption techniques fail to apply diffusion process in relation to an input image. In this chapter, a Fourier-Mellin moments based intertwining map is proposed to overcome the issue of less sensitivity towards an input image. Multi-objective Non-dominated sorting genetic algorithm (NSGA-II) based on Reinforcement learning (MNSGA-RL) is used for the optimization of Fourier-Mellin moments based intertwining logistic map. To further improve the computational speed, the proposed technique is also implemented in a parallel fashion using master-slave architecture.

## 7.2 Image encryption using Non-dominated sorting genetic algorithm

Image encryption technique based on Fourier-Mellin moments, intertwining logistic map, and MNSGA-RL known as IFIM is discussed in this section. It consists of nine main steps to encrypt an image (see Algorithm 16). The main steps of IFIM is depicted in Figure 7.1. The detailed description of steps are mentioned in the preceding subsections.

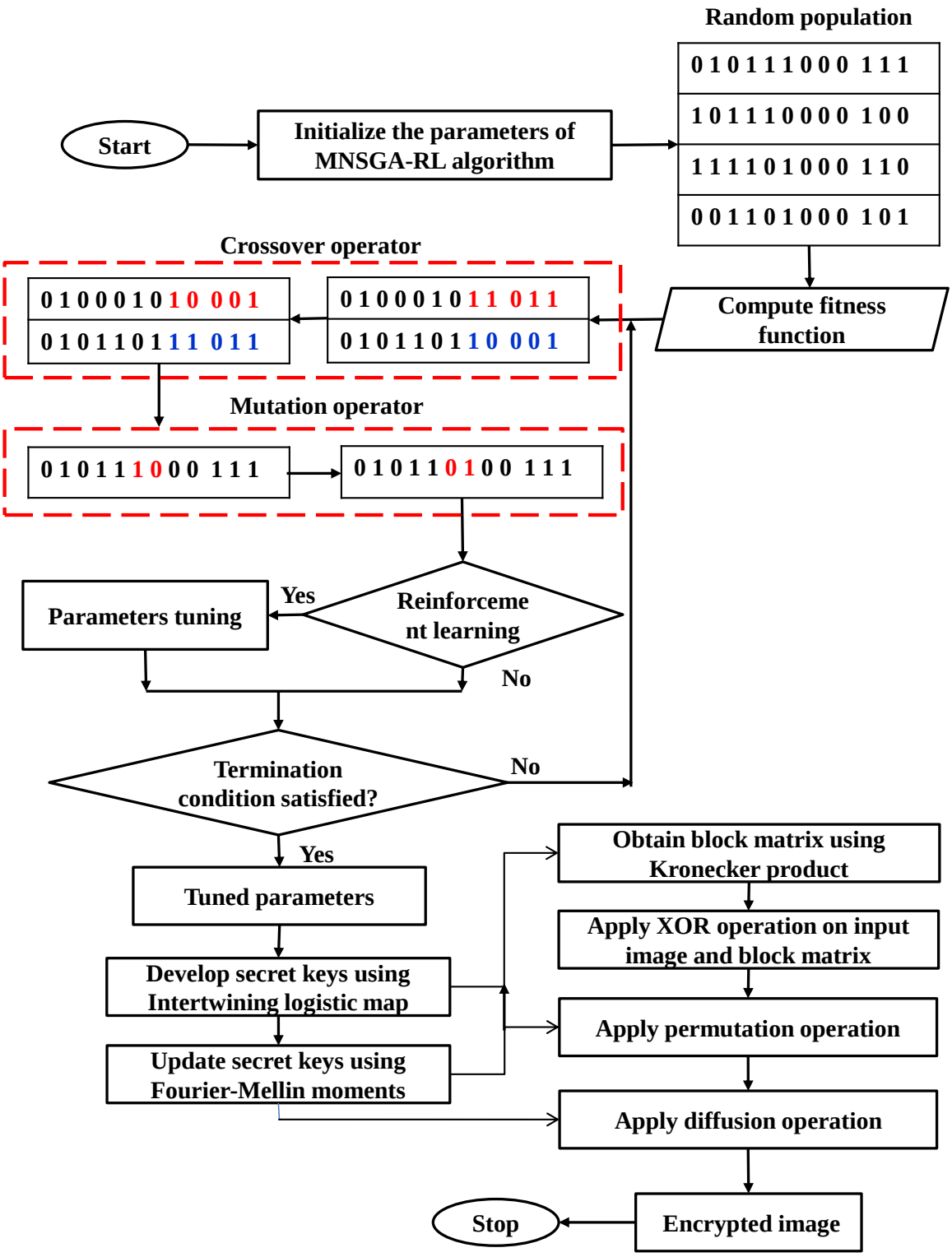



Figure 7.1: Diagrammatic flow of Fourier-Mellin moments, intertwining logistic map, and MNSGA-RL based image encryption technique

**Algorithm 16:** Non-dominated sorting genetic algorithm based image encryption

Step 1. Read input image $(I)$ and evaluate its width $(w)$ and height $(h)$.

Step 2. Apply Kronecker product on $x''$ of size $32 \times 32$ and Fourier- Mellin moments based matrix $c_p$ of size $8 \times 8$ to obtain block matrix $u$ of size $256 \times 256$ (see Section 7.2.1).

Step 3. Apply XOR operation on $I$ and $u$ to obtain pre-processed image $Q$.

Step 4. To tune initial parameters of intertwining logistic map, MNSGA-RL is used. Initially, random population are developed. Thereafter, fitness of each random solution is evaluated using multi-objective fitness function (see Section 7.2.4). Based upon evaluated fitness values, some best solutions are selected. Then, crossover and mutation operators are applied on them to obtain optimal parameters for intertwining logistic map.

Step 5. The tuned parameters of intertwining logistic map are updated using summation of pixels of $Q$. Then, these tuned parameters are used to generate three random sequences known as secret keys using Eq. (1.8).

Step 6. To make the secret keys more secure, Fourier-Mellin moments are used on these three keys. These keys are used to generate chaotic vectors, $G$ and $O$.

Step 7. Perform permutation operation on $Q$ using $G$ and $O$ to obtain permutated image $Q'$ (see Section 7.2.2).

Step 8. Again MNSGA-RL is used to obtain the tuned parameters for intertwining logistic map. Three secret keys $x_i$, $y_i$, and $z_i$ are generated using Eq. (1.8). Chaotic vector $R$ is produced from the combination of these three keys of length $wh + s$ .

Step 9. Perform diffusion operation on $Q'$ using $R$ to obtain the encrypted image $E$ (see Section 7.2.3).

### 7.2.1 Kronecker product

Kronecker product ($\otimes$) is an operation on two matrices of arbitrary size resulting in a block matrix [180]. It is generalization of outer product from vectors to matrices and gives the matrix of tensor product with respect to a standard choice of basis [181]. A block matrix $(u)$ is generated through Kronecker product. It is defined as:

$$u = x'' \otimes c_p \tag{7.1}$$

Here, $c_p$ represents Fourier-Mellin moments based matrix of size $8 \times 8$. $x''$ represents a chaotic sequence of size $32 \times 32$. Kronecker product $(u)$ is of order $256 \times 256$.

Let an Input image ($I$) of size $w \times h$. The preprocessed image ($Q$) can be obtained as:

$$Q = I \oplus u \tag{7.2}$$

where $\oplus$ represents XOR operation.

### 7.2.2 Permutation process

The pixel positions of an image is shuffled in permutation process to minimize the correlation between adjacent pixels [103]. But, the pixel values remain same. The summation for image $Q$ is calculated as:

$$\begin{aligned} r &= \sum_{i,j} Q_{i,j} \\ r_x &= \frac{r+1}{(wh+1) \times 255} \\ r_y &= \frac{r+2}{(wh+2) \times 255} \\ r_z &= \frac{r+3}{(wh+3) \times 255} \end{aligned} \tag{7.3}$$

where $r_x$, $r_y$, and $r_z$ are three parameters that generated from $r$. Thereafter, these parameters are used to update the initial state variables $(x_0, y_0, z_0)$ such as:

$$\begin{aligned} {x_0}' &= x_0 + r_x \mod 1 \\ {y_0}' &= y_0 + r_y \mod 1 \\ {z_0}' &= z_0 + r_z \mod 1 \end{aligned} \tag{7.4}$$

Here, $x'_0$, $y'_0$, and $z'_0$ represent updated initial state variables. Further, these variables are used to generate three chaotic sequences $x'_{i+1}$, $y'_{i+1}$, and $z'_{i+1}$ by utilizing Eq. (1.3).

Suppose that the chaotic sequences $x'_{i+1}$, $y'_{i+1}$, and $z'_{i+1}$ are represented as $G$, $O$, and $T$ chaotic vectors. These vectors change the pixel positions of an image $Q$. The permutation is carried out using following scheme:

$$\begin{aligned} G_i &= \lceil (G_i + T_i) \times 10^{14} \rceil \mod (h-1)+1 \\ O_j &= \lceil (O_j + T_j) \times 10^{14} \rceil \mod (w-1)+1 \end{aligned} \tag{7.5}$$

Here, $G_i$ is used to circularly shift the rows of image ($Q$) in right direction. $O_j$ is used to shift the columns of same image towards bottom. The column-wise shuffling is done after the row-wise shuffling. The final permutated image ($Q'$) is obtained.

### 7.2.3 Diffusion process

Diffusion process is carried out row-wise and column-wise. Initially, three chaotic sequences, *i.e.,* $x_{i+1}$, $y_{i+1}$, and $z_{i+1}$ are generated from $x_0$, $y_0$, and $z_0$ using Eq. (1.3). To avoid

the transient effect, first $s$ number of iterations are not used. Therefore, the sequences are started from $x_{s+i}$, $y_{s+i}$ and $z_{s+i}$. Then, vector $R$ is generated from the combination of these sequences whose length is $wh + s$. To map the pixel values of $R$ between $0$ and $255$, $R$ is computed as:

$$R_i = \lceil R_i \times 10^{14} \rceil \mod 256, \quad i = 1, 2, \ldots, wh + s \tag{7.6}$$

Now, diffusion operation is performed row-wise on permutated image $Q'$ as:

$$\begin{aligned} t_1 &= \sum_{i=1}^{m} Q'_i \\ t_2 &= t_1 \mod s + 1 \\ K_r &= R(1 + (i-1)m + t_2 : im + t_2) + t_2 \mod 256 \\ E'_i &= Q'_i + K_r \mod 256, \quad i = 1, 2, \ldots, m \end{aligned} \tag{7.7}$$

Here, $K_r$ represents the key-stream that depends on permutated image. $E'$ represents the final row-wise diffused image.

Next, column-wise diffusion is computed on $E'$ as:

$$\begin{aligned} t_1 &= \sum_{j=1}^{n} E'_j \\ t_2 &= t_1 \mod s + 1 \\ K_c &= R(1 + (j-1)n + t_2 : jn + t_2) + t_2 \mod 256 \\ E_j &= E'_j + K_c \mod 256, \quad j = 1, 2, \ldots, n \end{aligned} \tag{7.8}$$

where $K_c$ represents the key-stream that depends on row-wise diffused image. $E$ represents the final encrypted image that can obtained after row-wise and column-wise diffusion.

### 7.2.4 Multi-objective fitness function

IFIM uses multi-objective fitness function to evaluate the effectiveness of obtained solutions. The objective of fitness function is to minimize the correlation coefficient $(r_{x,y})$ and maximize three measures such as entropy (H(S)), Number of pixel change rates (NPCR), and Unified average change intensity values (UACI). The fitness function($f(z)$) is defined as:

$$\begin{aligned} \text{Maximize} \quad \text{f(z)} &= \frac{1}{3} \times \left( \frac{H(S)}{8} + \left(1 - r_{x,y}\right) + \frac{NPCR + UACI}{2} \right) \\ \text{subject to} \quad & H(S) \geq t_1 \end{aligned} \tag{7.9}$$

where $t_1$ denotes minimum required entropy value.

### 7.2.5 Decryption

The decryption process is same as encryption process but in reverse order. There is a need to send six secret keys (*i.e.,* $x_0$, $y_0$, $z_0$, $x_i$, $y_i$, and $z_i$), number of iterations that need to be avoided, *i.e.,* $s$, and $u$ to perform XOR operation.

## 7.3 Experimental results and discussion

This section validates the performance of proposed IFIM over five images and compares it with seven well-known techniques. Figure 7.2 shows the performance evaluation of IFIM. Figure 7.2 (a) and (b) show the input images and their histograms, respectively. Figure 7.2 (c) and (d) show the encrypted images and their respective histograms. The decrypted images are shown in Figure 7.2 (e). It is observed that the input and decrypted images are identical to each other. The histograms of encrypted images reveal that IFIM uniformly distribute the pixels.

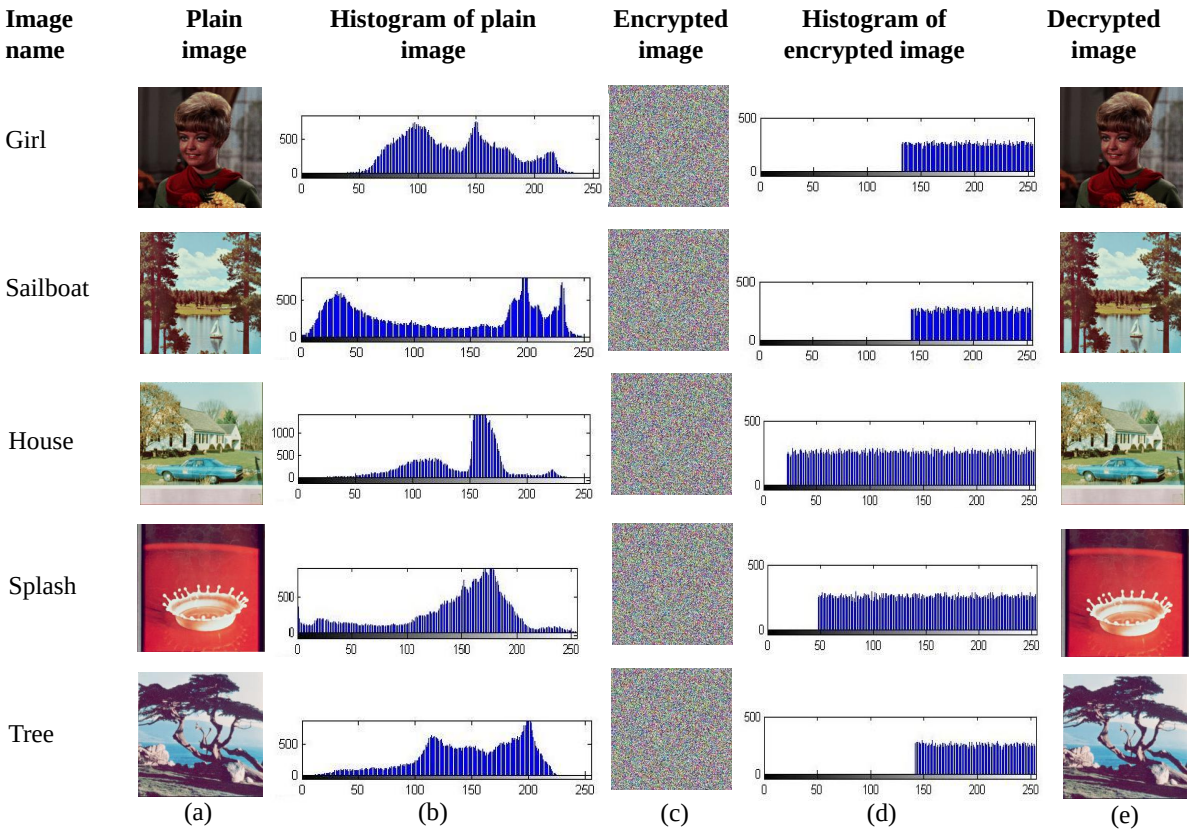



Figure 7.2: Performance evaluation of IFIM (a) Plain images, (b) Histograms of plain images, (c) Encrypted images, (d) Histograms of encrypted images, and (e) Decrypted images

### 7.3.1 Security analysis

In order to demonstrate the robustness of IFIM over different attacks, the security analysis has been done. The three well-known security analyses namely key space analysis, statistical analysis, and differential analysis have been performed. Comparisons have been drawn by considering the average of three color channels (*i.e.,* red, green, and blue) for each performance measure.

#### 7.3.1.1 Entropy

Table 7.1 shows the comparison of IFIM with the existing image encryption techniques in terms of entropy. It is observed that IFIM provides maximum entropy as compared to others. Therefore, pixels are uniformly distributed in the encrypted image and cannot be anticipated easily.

Table 7.1: Comparative analysis of IFIM in terms of entropy

| Technique | Girl | Sailboat | House | Splash | Tree |
|---|---|---|---|---|---|
| Arnold map | 7.9951 | 7.9945 | 7.9939 | 7.9784 | 7.9883 |
| Baker map | 7.9585 | 7.9651 | 7.9655 | 7.9685 | 7.9598 |
| Circle map | 7.9752 | 7.9749 | 7.9657 | 7.9760 | 7.9798 |
| Tent map | 7.9693 | 7.9753 | 7.9925 | 7.9719 | 7.9896 |
| Henon map | 7.8987 | 7.8992 | 7.8995 | 7.8990 | 7.8999 |
| Lorenz map | 7.9734 | 7.9702 | 7.9721 | 7.9823 | 7.9739 |
| Intertwining map | 7.9984 | 7.9989 | 7.9979 | 7.9990 | 7.9972 |
| IFIM | 7.9993 | 7.9994 | 7.9990 | 7.9995 | 7.9992 |

#### 7.3.1.2 Histogram analysis

Figure 7.2 (a) and (b) show the input images with their respective histograms, respectively. Figure 7.2 (c) and (d) show encrypted images and their respective histograms, respectively. It is observed from the histograms of encrypted images that the pixels are uniformly distributed. Therefore, it is hard to find any information from these encrypted images.

#### 7.3.1.3 Correlation analysis

Table 7.2 depicts horizontal, vertical, and diagonal correlation of IFIM. The correlation values of IFIM approach to zero which shows that there is no relation among the adjacent pixels. It is observed that attacker can hardly make any kind of relationship between pixels to break the algorithm.

Table 7.2: Correlation coefficient of color images using IFIM

| Images | Plain image | | | Cipherd image | | |
|---|---|---|---|---|---|---|
| | Horizontal | Vertical | Diagonal | Horizontal | Vertical | Diagonal |
| Girl | 0.9446 | 0.9649 | 0.9438 | −0.0012 | 0.0015 | 0.0022 |
| Sailboat | 0.9556 | 0.9456 | 0.9822 | 0.0002 | −0.0025 | 0.0011 |
| House | 0.9701 | 0.9455 | 0.8669 | 0.0009 | −0.0019 | 0.0020 |
| Splash | 0.9243 | 0.9367 | 0.9081 | −0.0008 | 0.0016 | 0.0034 |
| Tree | 0.9714 | 0.9758 | 0.9965 | 0.0018 | 0.0038 | 0.0058 |

#### 7.3.1.4 Key space analysis

In IFIM, six keys are used to encrypt the image. The three keys (*i.e.,* $x'_0$, $y'_0$, and $z'_0$) are used to permute the input image. The three keys (*i.e.,* $x_i$, $y_i$, and $z_i$) are used to diffuse the permuted image. The key space of IFIM can be reached approximately to $10^{84}$, if precision is set to $10^{-14}$. Therefore, IFIM provides a huge key space to resist the brute-force attacks.

#### 7.3.1.5 Differential attack analysis

Tables 7.3 and 7.4 show the comparison of IFIM with existing techniques in terms of NPCR and UACI, respectively. The results reveal that IFIM provides better values of NPCR and UACI as compared to others. The maximum value of NPCR is $0.9970$ (*i.e.,* $99.7\%$) and UACI is $0.3362$ (*i.e.,* $33.62\%$). This indicates that when only one pixel is changed in an original image, IFIM will generate a completely different encrypted image.

Table 7.3: Comparative analysis of IFIM in terms of NPCR

| Technique | Girl | Sailboat | House | Splash | Tree |
|---|---|---|---|---|---|
| Arnold map | 0.9949 | 0.9950 | 0.9951 | 0.9950 | 0.9941 |
| Baker map | 0.9940 | 0.9621 | 0.9620 | 0.9623 | 0.9622 |
| Circle map | 0.9942 | 0.9951 | 0.9947 | 0.9950 | 0.9949 |
| Tent map | 0.9950 | 0.9943 | 0.9949 | 0.9952 | 0.9953 |
| Henon map | 0.9935 | 0.9945 | 0.9932 | 0.9949 | 0.9950 |
| Lorenz map | 0.9921 | 0.9931 | 0.9923 | 0.9929 | 0.9934 |
| Intertwining map | 0.9926 | 0.9940 | 0.9936 | 0.9930 | 0.9929 |
| IFIM | 0.9968 | 0.9970 | 0.9969 | 0.9962 | 0.9965 |

Table 7.4: Comparative analysis of IFIM using UACI

| Technique | Girl | Sailboat | House | Splash | Tree |
|---|---|---|---|---|---|
| Arnold map | 0.3339 | 0.3315 | 0.3314 | 0.3314 | 0.3314 |
| Baker map | 0.3319 | 0.3351 | 0.3353 | 0.3350 | 0.3313 |
| Circle map | 0.3330 | 0.3355 | 0.3353 | 0.3353 | 0.3351 |
| Tent map | 0.3354 | 0.3353 | 0.3355 | 0.3351 | 0.3339 |
| Henon map | 0.3355 | 0.3359 | 0.3351 | 0.3354 | 0.3351 |
| Lorenz map | 0.3331 | 0.3339 | 0.3354 | 0.3351 | 0.3350 |
| Intertwining map | 0.3311 | 0.3319 | 0.3331 | 0.3334 | 0.3330 |
| IFIM | 0.3362 | 0.3359 | 0.3357 | 0.3356 | 0.3354 |

## 7.4 Parallel non-dominated sorting genetic algorithm

Although, IFIM outperforms the existing techniques. But, it suffers from poor computational speed especially for images with high resolution. To resolve this issue, IFIM is implemented in parallel fashion by designing a master-slave environment. Initially, the execution time analysis of IFIM is done for determining the computationally expensive operations. Thereafter, IFIM operations are divided into master and slave jobs. Message passing interface (MPI) is used for intercommunication between master and slave nodes.

### 7.4.1 Theoretical analysis

To design an efficient parallel IFIM, the execution time analysis of sequential NSGA is considered. It decomposes the number of operations into two parts, *i.e.,* operations which take more time and operations which take lesser time. It is observed that selection, crossover, and mutation are computationally intensive. Whereas, population initialization, fitness evaluation, non-dominated ranking, crowding distance, and termination criteria evaluation are not computationally intensive in nature.

### 7.4.2 General scheme

To exploit the parallel environment, master-slave based NSGA is discussed in this subsection. It contains a set of sub jobs ($s_l$). MPI jobs placed at different processing elements ($PE$), introducing intra-$PE$ communication by defining MPI tasks ($n_t$) per job, population ($P$) and maximum generations ($m_g$).

Algorithm 17 shows that one job act as a master task and responsible for dividing the given problem (*i.e.,* $N_p$) into a number of tasks and it also conquers the results from the other $PEs$. Afterward, it may recall slave $PEs$ or return the final solution based upon the stopping criteria. The slaves, *i.e.,* remaining $PEs$ except master $PE$ are responsible

for applying various operators of NSGA such as selection, crossover, and mutation. Algorithm 17 shows a coarse-grained technique that spawns an MPI task pool by considering $\#processMPI$. Figure 7.3 shows the flowchart of parallel IFIM.

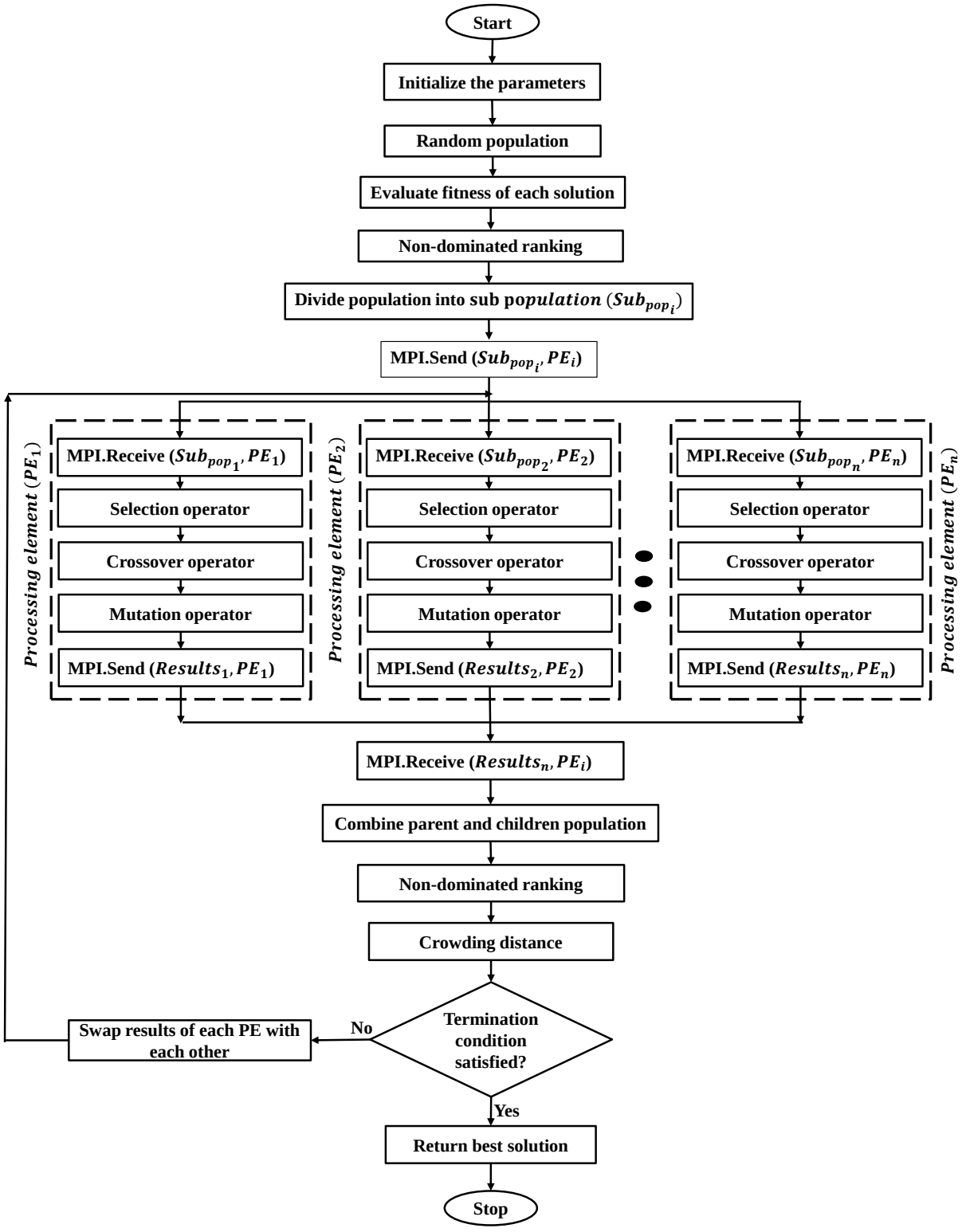



Figure 7.3: Flowchart of parallel non-dominated sorting genetic algorithm

**Algorithm 17:** Parallel non-dominated sorting genetic algorithm

```
1  Initialize MPI
2  #process MPI tasks (n_t)
3  for i = 1 to m_g do
4      if master job then
5          Do master jobs (P)
6          #process MPI single
7          MPI Send (P.M, jobsNumber/s_l, l) /*∀ l: l=2 to s_l*/
8      else
9          #process MPI single
10         MPI Receive (Q.M, jobsNumber/s_l, master)
11     end
12     /* ∀ l: l=1 to s_l */
13     Execute slave jobs (P)
14     if master job then
15         #process MPI single
16         MPI Receive (Results, jobsNumber/s_l, l) /*∀ l: l=2 to s_l*/
17         ParetoFront ← updateParetoFront(Results)
18     else
19         #process MPI single MPI Send (Results, jobsNumber/s_l, master)
20     end
21 end
```

These tasks are utilized according to the computational requirements of the algorithm.

#### 7.4.2.1 Master jobs

The master node is responsible for generating the initial population ($P$). Thereafter, it evaluates the fitness value of each population and ranks $P$ on the basis of non-domination values. Afterward, it divides $P$ into tasks (*i.e.,* threads). It also combines the offspring population ($Q$) that obtained from the slave jobs with $P$. To obtain the best solution, master node applies non-dominated sorting and crowding distance on combined population (*i.e.,* $C_p = P \cup Q$). Master tasks are discussed in Algorithm 18.

The master will transmit the obtained non-dominated solutions to slaves. The master job changes its role to slave to avoid idle resources while performing slave computations (see Algorithm 17).

**Algorithm 18:** Master jobs

**Input:** Population $P$ and maximum generations $m_g$

**Output:** Updated population $P$

Generate initial population $P$
Evaluate fitness of $P$
Evaluate non-dominated-sort($P$)
#process MPI /* Call slaves */
**for** *k = 1 to* $N_p$ **do**
    $Q_k$.M ← applyNSGAOperators(P, crossoverRate, mutationRate)
**end**
Evaluate the fitness of $Q$
$C_p$ ← P∪ Q
Fronts← sortCombined$P(C_p)$
Evaluate crowdingDistance(Fronts)
P ← updateParent $P$

#### 7.4.2.2 Slave jobs

Slave jobs (as described in Algorithm 19) are used to do various computationally expensive jobs/tasks in a parallel fashion. These tasks are allocated to slave jobs by the master. Each slave node completes its job. Thereafter, the obtained results are sent back to the master.

**Algorithm 19:** Slave Jobs

#process MPI for schedule
**for** *k = 1 to jobsNumber/*$s_l$ **do**
    $P_k.s$ ← applySelection($P_k$.M, dataset$_{idThread}$)
    $P_k.c$ ← applyCrossover($P_k.s$, dataset$_{idThread}$)
    $Q_k.m$ ← applyMutation($P_k.c$, dataset$_{idThread}$)
**end**

#### 7.4.2.3 Intercommunication

Intercommunication is used in master-slave model to reduce the communication overheads. Intercommunication is required during job allocation and conquering the results. Whereas, slave computations require the communication of $N_p/s_l$ per job. Therefore, the synchronization is considered at a task level. It takes place at implicit barriers generated by $\#processMPI$ for directives. Thus, load balance problems may occur when loops are implemented in parallel fashion with a dynamic workload. To overcome this issue, different MPI scheduling techniques can be considered. Thereafter, inter-job communication is used to manage an efficient distribution of jobs and tasks.

## 7.5 Computational speed analysis

Nine slave nodes are utilized to implement IFIM in a parallel manner. Table 6.11 depicts the computation time analysis of IFIM and existing image encryption techniques. From Table 7.5, it is found that the parallel version of IFIM provides significantly good computational speed as compared to the existing image encryption techniques. Parallel IFIM achieves a significantly good computational speed when the size of input images increases. In Table 7.5, Seq. and Par. represent sequential and parallel version of IFIM, respectively.

Table 7.5: Computational speed analysis of the encryption process

| Image | Size | WDICA | GA | DHS | GA | MNSGA | IFIM (Seq.) | IFIM (Par.) |
|---|---|---|---|---|---|---|---|---|
| Art | $256 \times 256$ | 23.9 | 24.7 | 22.8 | 26.4 | 23.4 | 21.2 | 17.6 |
| | $512 \times 512$ | 61.3 | 71.5 | 62.3 | 63.6 | 58.2 | 47.9 | 28.7 |
| | $1024 \times 1024$ | 749.3 | 687.2 | 726.6 | 715.4 | 625.3 | 511.8 | 321.4 |
| | $2048 \times 2048$ | 4781.2 | 4943.6 | 4978.7 | 5119.1 | 4997.5 | 3941.7 | 1397.9 |
| Cone | $256 \times 256$ | 24.7 | 25.3 | 26.3 | 27.4 | 22.4 | 22.0 | 18.3 |
| | $512 \times 512$ | 60.4 | 69.6 | 64.5 | 66.7 | 57.6 | 46.8 | 26.9 |
| | $1024 \times 1024$ | 753.4 | 694.9 | 769.1 | 722.3 | 614.2 | 565.2 | 310.8 |
| | $2048 \times 2048$ | 4657.5 | 4844.3 | 4729.4 | 5028.3 | 5003.9 | 3978.2 | 1348.6 |
| Teddy | $256 \times 256$ | 25.2 | 28.1 | 26.9 | 28.5 | 27.3 | 23.5 | 19.0 |
| | $512 \times 512$ | 59.8 | 57.5 | 60.3 | 62.1 | 55.2 | 48.4 | 23.3 |
| | $1024 \times 1024$ | 645.1 | 675.4 | 680.3 | 639.8 | 607.9 | 533.9 | 341.7 |
| | $2048 \times 2048$ | 4825.4 | 4869.1 | 4911.6 | 4877.5 | 4763.3 | 4533.1 | 1356.2 |
| Book | $256 \times 256$ | 22.9 | 25.5 | 28.3 | 26.9 | 24.6 | 21.9 | 16.9 |
| | $512 \times 512$ | 58.9 | 56.7 | 63.4 | 59.8 | 66.1 | 45.4 | 25.5 |
| | $1024 \times 1024$ | 736.4 | 750.2 | 789.6 | 689.9 | 658.1 | 521.3 | 299.7 |
| | $2048 \times 2048$ | 4745.6 | 4737.3 | 4897.5 | 4755.2 | 4711.6 | 4189.7 | 1398.9 |

## 7.6 Summary

In this chapter, an IFIM has been designed. NSGA-II has been used to tune the initial parameters of the intertwining logistic map. To improve the computational speed, NSGA-II has been implemented in a parallel fashion using master-slave architecture. Fourier-Mellin moments based matrix has been used to improve the secret keys obtained from the intertwining logistic map. The permutation and diffusion operations have been carried out on an input image using these secret keys to encrypt the images. Experimental results have shown that IFIM provides encryption and decryption results at good computational speed as compared to the existing meta-heuristic based image encryption techniques.

# Chapter 8

# Conclusions and future work

This chapter concludes the thesis by explaining the outcome of each chapter. Future directions of the research work that can be carried out as part of extension to present work.

## 8.1 Conclusions

Images have become one of the popular data formats across the world to share the potential information. With the advancement in internet technology, these images have turned out to be the primary way to communicate secret digital information. Therefore, the security of these images requires a crucial attention. With this aim, many image encryption techniques have been designed to secure these images in an efficient manner.

Although many image encryption techniques have been designed by various researchers. Some of them are proven to have insufficient security against various security attacks such as brute-force, known-plaintext, ciphertext, chosen plaintext attack, *etc.* High-dimensional chaotic based encryption techniques possess complicated chaotic behavior and are hard to predict their chaotic revolutions. However, these are suffer from key sensitivity, parameter tuning, and poor computational speed. In view of this, various image encryption techniques have been designed in this research work.

Initially, an image encryption technique that uses GA, beta chaotic map, and nonsubsampled contourlet transform known as IGN has been proposed. Three performance measures such as entropy, the number of pixel change rates, and unified average change intensity values of an encrypted image have been used to design the multi-objective fitness function. IGN has an ability to tune the required parameters of beta chaotic map for a secure key generation. The beta chaotic map is able to generate different secret key for every input image. Hence, it makes hard for an attacker to discover the secret key. IGN has been tested on ten well-known benchmark images. It has been observed that IGN is computationally faster than the other existing image encryption techniques. It also provides significant quality of decrypted images. To test the security of IGN, various experiments

have been carried out such as statistical attack analysis, differential attack analysis, secret key analysis, noise attack analysis, and occlusion attack analysis. The experimental results reveal that IGN provides better performance than the other techniques.

Thereafter, a novel differential evolution based image encryption technique (*i.e.,* IDN) has been proposed. IDN uses Arnold transform to scramble the input image. NSCT is then utilized to decompose the scrambled image into sub-bands. A beta chaotic map is used to generate a secret key which further utilized to encrypt the sub-bands. The differential evolution is utilized for fine-tuning of beta chaotic map . The mean improvement of IGN in terms of entropy, NPCR, UACI, PSNR, and MAE are 0.22%, 0.09%, 0.10%, 20.5% (dB), and 9.9%, respectively. The mean reduction of the proposed technique in case of correlation coefficient is 2.9%. Experimental results reveal that IDN outperforms the existing competitive image encryption techniques.

To improve the robustness and security of intertwining logistic map based encryption, an IIMA is designed and implemented. IIMA utilized memetic differential evolution and Arnold transform to create confusion among the position of pixels. The effectiveness of IIMA has been tested on five color images. On comparing the results of IIMA with others, it has been observed that the mean improvement of IIMA over others in terms of entropy, NPCR, UACI, and PSNR are 0.051 %, 0.065 %, 0.097 %, and 2.17 % (dB), respectively. The correlation coefficient of IIMA is reduced by 0.8%. The results reveal that IIMA provides higher efficiency and security of images among all competitive approaches.

Thereafter, an efficient image encryption technique based on SHA-3, adaptive differential evolution, and Lorenz-like chaotic system so-called ISAL has been proposed. The main novelty of ISAL is to improve the parameters selection of Lorenz-like chaotic system. It has been achieved by utilizing SHA-3 and adaptive differential evolution. ISAL sensitive towards an input image and secret keys. It adaptively selects input parameters of Lorenz-like chaotic system. The different types of security analyses such as statistical, differential attack, noise attack, and image enhancement have been performed on ISAL. The comparative results reveal that ISAL provides better encryption results than the others. ISAL is able to resist against statistical, differential, noise, and image enhancement attacks.

To overcome the issue of low sensitivity and parameter tuning with the intertwining logistic map, a novel image encryption technique so-called IFIM is designed. In IFIM, MNSGA-RL has used to optimize the initial parameters of intertwining logistic map. To overcome the issue of low sensitivity, the optimized parameters of intertwining logistic map are updated using the summation of pixel values of an input image. Fourier-Mellin moment based matrix is implemented to modify the intertwining logistic map based secret keys. Thereafter, these secret keys have been used to carry out permutation and diffusion operations on the input image to obtain the encrypted image. The performance of IFIM has been evaluated on five well-known benchmark images and also compared with seven well-known existing encryption techniques. The mean improvement of IFIM in terms of entropy, NPCR, and UACI is 2.19%, 1.09%, and 1.82%, respectively. The mean reduction

of IFIM in case of a correlation coefficient is 2.9%. Extensive security analyses demonstrate high level of security of IFIM and show its robustness against various types of attacks. To improve the computational speed, IFIM has been implemented in a parallel fashion using master-slave environment. Initially, the execution time analysis of IFIM has been done to evaluate the computationally expensive operations. Thereafter, IFIM operations have been divided into master and slave jobs. Message Passing Interface (MPI) has been used for intercommunication between master and slave nodes. The simulation results demonstrate that parallel IFIM provides a significant improvement in computational speed as compared to the existing techniques and sequential IFIM.

In this research work, we have proposed five different image encryption techniques (*i.e.,* IGN, IDN, IIMA, ISAL, and IFIM). Extensive experiments have been carried out to evaluate the effectiveness of proposed image encryption techniques. It has been found that the proposed image encryption techniques can effectively resist differential, statistical, noise, and chosen-plaintext attacks. Therefore, the proposed image encryption techniques are more suitable for a real-time end to end communication of digital images.

## 8.2 Scope for future work

The work presented in this thesis can be extended in the following future research directions.

i In near future, we may consider some recently designed and implemented techniques in the literature such as Quantum inspired evolutionary technique, differential evolution with composite trial vector generation, cellular memetic algorithm, backtracking search optimization algorithm, etc. These techniques can be applied to the proposed techniques to obtain more significant results at good computational speed.

ii In this research work, we have not considered the use of compressive sensing and optical domain to design the proposed techniques. Therefore, in near future, these techniques can be used to improve the computational speed of the proposed techniques.

iii The proposed image encryption techniques are more appropriate for high resolution images only. These techniques can be implemented in low computing devices, but, may suffer from poor computational speed. Therefore, in order to utilize the proposed techniques in low computing devices, in near future, we may consider compressive sensing and optical domain techniques. Also, we may relax evolutionary techniques to enhance the computational speed further.

iv This research work has overcome the issue of parameter tuning with existing variants of chaotic maps. Therefore, in near future, we can propose a novel chaotic map to improve the performance of the proposed techniques.

v The proposed techniques, in this thesis, can be tested against multiplicative security attacks and cryptanalysis.

vi To hide the existence of secret images, steganography can be used in the proposed techniques.

# List of publications

1. Manjit Kaur and Vijay Kumar, "A Comprehensive Review on Image Encryption Techniques", Archives of Computational Methods in Engineering, Springer, pp. 1-29, 2019. **[SCI Indexed, Impact Factor 7.242]**

2. Manjit Kaur and Vijay Kumar, "Color Image Encryption Technique using Differential Evolution in Nonsubsampled Contourlet Transform Domain", IET Image Processing, vol. 12, no. 7, pp. 1273-1283, 2018. **[SCI Indexed, Impact Factor 2.004]**

3. Manjit Kaur and Vijay Kumar, "Efficient image encryption method based on improved Lorenz chaotic system", Electronics Letters, IET, vol. 54, no. 9, pp. 562-564, 2018. **[SCI Indexed, Impact Factor 1.343]**

4. Manjit Kaur and Vijay Kumar, "Adaptive differential evolution based Lorenz chaotic system for image encryption", Arabian Journal for Science and Engineering, Springer, vol. 43, no. 12, pp. 8127?8144, 2018. **[SCI Indexed, Impact Factor 1.518]**

5. Manjit Kaur and Vijay Kumar, "Beta chaotic map based image encryption using genetic algorithm", International Journal of Bifurcation and Chaos, vol. 28, no. 11, pp. 1850132, 2018. **[SCI Indexed, Impact Factor 1.501]**

6. Manjit Kaur, Vijay Kumar, and Li Li "Color image encryption approach based on memetic differential evolution", Neural Computing And Applications, pp. 1-18, 2018. **[SCI Indexed, Impact Factor 4.664]**

7. Manjit Kaur and Vijay Kumar, "Fourier-Mellin moment-based intertwining map for image encryption", Modern Physics Letters B, vol. 32, no. 09, pp. 1850115, 2018. **[SCI Indexed, Impact Factor 0.834]**

8. Manjit Kaur and Vijay Kumar, "Parallel Non-dominated Sorting Genetic Algorithm-II based Image Encryption Technique", Imaging Science Journal, vol. 66, no. 8, pp. 453-462, 2018. **[SCI Indexed, Impact Factor 0.451]**

# Bibliography


[1] M. Khan and T. Shah, "An efficient chaotic image encryption scheme," *Neural Computing and Applications*, vol. 26, no. 5, pp. 1137–1148, 2015.

[2] S. Sabharwal and M. Aggarwal, "Test set generation for pairwise testing using genetic algorithms," *JIPS (Journal of Information Processing Systems)*, vol. 13, no. 5, pp. 1089–1102, 2017.

[3] J. Ahmad, M. A. Khan, F. Ahmed, and J. S. Khan, "A novel image encryption scheme based on orthogonal matrix, skew tent map, and xor operation," *Neural Computing and Applications*, pp. 1–11, 2017.

[4] X. Chai, Z. Gan, K. Yuan, Y. Chen, and X. Liu, "A novel image encryption scheme based on dna sequence operations and chaotic systems," *Neural Computing and Applications*, pp. 1–19, 2017.

[5] T. Sivakumar and R. Venkatesan, "A novel image encryption using calligraphy based scan method and random number.," *KSII Transactions on Internet & Information Systems*, vol. 9, no. 6, 2015.

[6] G. Gu and J. Ling, "A fast image encryption method by using chaotic 3d cat maps," *Optik-International Journal for Light and Electron Optics*, vol. 125, no. 17, pp. 4700–4705, 2014.

[7] S. Awasthi and Y. Singh, "Biased contribution index: a new faster convergent index to maintain the fairness in peer-to-peer networks," *Electronics Letters*, vol. 54, pp. 1174–1176(2), October 2018.

[8] E. Mohebi and A. Bagirov, "A convolutional recursive modified self organizing map for handwritten digits recognition," *Neural Networks*, vol. 60, pp. 104 – 118, 2014.

[9] S. Sabharwal, P. Bansal, and N. Mittal, "Construction of t-way covering arrays using genetic algorithm," *International Journal of System Assurance Engineering and Management*, vol. 8, no. 2, pp. 264–274, 2017.

[10] A. K. Singh, B. Kumar, S. K. Singh, S. Ghrera, and A. Mohan, "Multiple watermarking technique for securing online social network contents using back propagation neural network," *Future Generation Computer Systems*, vol. 86, pp. 926 – 939, 2018.

[11] S. Seifollahi, A. Bagirov, R. Layton, and I. Gondal, "Optimization based clustering algorithms for authorship analysis of phishing emails," *Neural Processing Letters*, vol. 46, no. 2, pp. 411–425, 2017.

[12] A. Ono and T. Kohda, "Solvable three-dimensional rational chaotic map defined by jacobian elliptic functions," *International Journal of Bifurcation and Chaos*, vol. 17, no. 10, pp. 3645–3650, 2007.

[13] G. Chen, D. Zhang, Q. Chen, and D. Zhou, "The characteristic of different chaotic sequences for compressive sensing," in *Image and Signal Processing (CISP), 2012 5th International Congress on*, pp. 1475–1479, IEEE, 2012.

[14] A. Umamageswari and G. Suresh, "Security in medical image communication with arnold's cat map method and reversible watermarking," in *Circuits, Power and Computing Technologies (ICCPCT), 2013 International Conference on*, pp. 1116–1121, IEEE, 2013.

[15] M. Salleh, S. Ibrahim, and I. F. Isnin, "Enhanced chaotic image encryption algorithm based on baker's map," in *Circuits and Systems, 2003. ISCAS'03. Proceedings of the 2003 International Symposium on*, vol. 2, pp. II–II, IEEE, 2003.

[16] D. Sadhya and S. K. Singh, "Design of a cancelable biometric template protection scheme for fingerprints based on cryptographic hash functions," *Multimedia Tools and Applications*, vol. 77, no. 12, pp. 15113–15137, 2018.

[17] H. Hu, L. Liu, and N. Ding, "Pseudorandom sequence generator based on the chen chaotic system," *Computer Physics Communications*, vol. 184, no. 3, pp. 765–768, 2013.

[18] S. Saponara, L. Fanucci, S. Marsi, and G. Ramponi, "Algorithmic and architectural design for real-time and power-efficient retinex image/video processing," *Journal of real-time image processing*, vol. 1, no. 4, pp. 267–283, 2007.

[19] D. Chattopadhyay, M. Mandal, and D. Nandi, "Symmetric key chaotic image encryption using circle map," *Indian Journal of Science and Technology*, vol. 4, no. 5, pp. 593–599, 2011.

[20] N. Singha and Y. N. Singh, "Optimal capacity partitioning in homogeneous p2p network," *IEEE Communications Letters*, vol. 22, pp. 1354–1357, July 2018.

[21] M. C. Shastry, N. Nagaraj, and P. G. Vaidya, "The b-exponential map: A generalization of the logistic map, and its applications in generating pseudo-random numbers," *arXiv preprint cs/0607069*, 2006.

[22] T. Gao and Z. Chen, "A new image encryption algorithm based on hyper-chaos," *Physics Letters A*, vol. 372, no. 4, pp. 394–400, 2008.

[23] P. Jindal and B. Singh, "Performance evaluation of security-throughput tradeoff with channel adaptive encryption," *International Journal of Computer Network and Information Security*, vol. 5, no. 1, p. 49, 2013.

[24] C. Wei-Bin and Z. Xin, "Image encryption algorithm based on henon chaotic system," in *Image Analysis and Signal Processing, 2009. IASP 2009. International Conference on*, pp. 94–97, IEEE, 2009.

[25] N. K. Pareek, V. Patidar, and K. K. Sud, "Image encryption using chaotic logistic map," *Image and vision computing*, vol. 24, no. 9, pp. 926–934, 2006.

[26] W. Zhen, H. Xia, L. Yu-Xia, and S. Xiao-Na, "A new image encryption algorithm based on the fractional-order hyperchaotic lorenz system," *Chinese Physics B*, vol. 22, no. 1, p. 010504, 2013.

[27] A. Kadir, A. Hamdulla, and W.-Q. Guo, "Color image encryption using skew tent map and hyper chaotic system of 6th-order cnn," *Optik-International Journal for Light and Electron Optics*, vol. 125, no. 5, pp. 1671–1675, 2014.

[28] R. Enayatifar, H. J. Sadaei, A. H. Abdullah, M. Lee, and I. F. Isnin, "A novel chaotic based image encryption using a hybrid model of deoxyribonucleic acid and cellular automata," *Optics and Lasers in Engineering*, vol. 71, pp. 33–41, 2015.

[29] H. Zeng, K.-K. Ma, and C. Cai, "Fast mode decision for multiview video coding using mode correlation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 11, pp. 1659–1666, 2011.

[30] R. Hamza and F. Titouna, "A novel sensitive image encryption algorithm based on the zaslavsky chaotic map," *Information Security Journal: A Global Perspective*, vol. 25, no. 4-6, pp. 162–179, 2016.

[31] P. Jindal and B. Singh, "Quantitative analysis of the security performance in wireless lans," *Journal of King Saud University-Computer and Information Sciences*, vol. 29, no. 3, pp. 246–268, 2017.

[32] R. Zahmoul, R. Ejbali, and M. Zaied, "Image encryption based on new beta chaotic maps," *Optics and Lasers in Engineering*, vol. 96, pp. 39–49, 2017.

[33] S. Saponara, L. Fanucci, S. Marsi, G. Ramponi, D. Kammler, and E. M. Witte, "Application-specific instruction-set processor for retinex-like image and video processing," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 54, no. 7, pp. 596–600, 2007.

[34] I. S. Sam, P. Devaraj, and R. Bhuvaneswaran, "An intertwining chaotic maps based image encryption scheme," *Nonlinear Dynamics*, vol. 69, no. 4, pp. 1995–2007, 2012.

[35] H. Gao, Y. Zhang, S. Liang, and D. Li, "A new chaotic algorithm for image encryption," *Chaos, Solitons & Fractals*, vol. 29, no. 2, pp. 393–399, 2006.

[36] W. Xingyuan, F. Le, W. Shibing, C. Zhang, and Z. Yingqian, "Spatiotemporal chaos in coupled logistic map lattice with dynamic coupling coefficient and its application in image encryption," *IEEE Access*, vol. 6, pp. 39705–39724, 2018.

[37] M. Dong, H. Zeng, J. Chen, C. Cai, and K.-K. Ma, "Multiple description video coding based on adaptive data reuse," *Journal of Visual Communication and Image Representation*, vol. 38, pp. 378 – 385, 2016.

[38] X.-Y. Wang, P. Li, Y.-Q. Zhang, L.-Y. Liu, H. Zhang, and X. Wang, "A novel color image encryption scheme using dna permutation based on the lorenz system," *Multimedia Tools and Applications*, vol. 77, no. 5, pp. 6243–6265, 2018.

[39] N. Sharma, I. Saini, A. Yadav, and P. Singh, "Phase-image encryption based on 3d-lorenz chaotic system and double random phase encoding," *3D Research*, vol. 8, no. 4, p. 39, 2017.

[40] L. Huang, X. Wang, and G. Sun, "Design and circuit simulation of the new lorenz chaotic system," in *Systems and Control in Aeronautics and Astronautics (ISSCAA), 2010 3rd International Symposium on*, pp. 1443–1447, IEEE, 2010.

[41] M. A. El-Sayed, M. Hassaballah, and M. A. Abdel-Latif, "Identity verification of individuals based on retinal features using gabor filters and svm," *Journal of Signal and Information Processing*, vol. 7, no. 01, p. 49, 2016.

[42] A. A. Alzaidi, M. Ahmad, M. N. Doja, E. A. Solami, and M. M. S. Beg, "A new 1d chaotic map and$\beta$-hill climbing for generating substitution-boxes," *IEEE Access*, vol. 6, pp. 55405–55418, 2018.

[43] M. Hassaballah, M. Makky, and Y. B. Mahdy, "A fast fractal image compression method based entropy," *ELCVIA Electronic Letters on Computer Vision and Image Analysis*, vol. 5, no. 1, pp. 30–40, 2005.

[44] W. Wei, Y. Li, A. J. Deegan, and R. K. Wang, "Mapping and quantitating penetrating vessels in cortical brain using eigen-decomposition of oct signals and subsequent principal component analysis," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 25, no. 1, pp. 1–9, 2019.

[45] L. Huang, D. Shi, and J. Gao, "The design and its application in secure communication and image encryption of a new lorenz-like system with varying parameter," *Mathematical Problems in Engineering*, vol. 2016, 2016.

[46] R. Enayatifar, A. H. Abdullah, and I. F. Isnin, "Chaos-based image encryption using a hybrid genetic algorithm and a DNA sequence," *Optics and Lasers in Engineering*, vol. 56, pp. 83–93, 2014.

[47] A. H. Abdullah, R. Enayatifar, and M. Lee, "A hybrid genetic algorithm and chaotic function model for image encryption," *AEU-International Journal of Electronics and Communications*, vol. 66, no. 10, pp. 806–816, 2012.

[48] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[49] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE transactions on evolutionary computation*, vol. 15, no. 1, pp. 4–31, 2011.

[50] L. Yliniemi and K. Tumer, "Multi-objective multiagent credit assignment in reinforcement learning and nsga-ii," *Soft Computing*, vol. 20, no. 10, pp. 3869–3887, 2016.

[51] T. C. Bora, L. Lebensztajn, and L. D. S. Coelho, "Non-dominated sorting genetic algorithm based on reinforcement learning to optimization of broad-band reflector antennas satellite," *IEEE Transactions on Magnetics*, vol. 48, no. 2, pp. 767–770, 2012.

[52] D. Jia, G. Zheng, and M. K. Khan, "An effective memetic differential evolution algorithm based on chaotic local search," *Information Sciences*, vol. 181, no. 15, pp. 3175–3187, 2011.

[53] Q. Lin, Q. Zhu, P. Huang, J. Chen, Z. Ming, and J. Yu, "A novel hybrid multi-objective immune algorithm with adaptive differential evolution," *Computers & Operations Research*, vol. 62, pp. 95–111, 2015.

[54] A. Belazi, A. A. A. El-Latif, and S. Belghith, "A novel image encryption scheme based on substitution-permutation network and chaos," *Signal Processing*, vol. 128, pp. 155–170, 2016.

[55] X. Chai, Y. Chen, and L. Broyde, "A novel chaos-based image encryption algorithm using dna sequence operations," *Optics and Lasers in Engineering*, vol. 88, pp. 197 – 213, 2017.

[56] G. Zhao, G. Chen, J. Fang, and G. Xu, "Block cipher design: Generalized single-use-algorithm based on chaos," *Tsinghua Science and Technology*, vol. 16, no. 2, pp. 194–206, 2011.

[57] F. E. A. El-Samie, H. E. H. Ahmed, I. F. Elashry, M. H. Shahieen, O. S. Faragallah, E.-S. M. El-Rabaie, and S. A. Alshebeili, *Image encryption: a communication perspective*. CRC Press, 2013.

[58] X. Chai, Z. Gan, K. Yang, Y. Chen, and X. Liu, "An image encryption algorithm based on the memristive hyperchaotic system, cellular automata and {DNA} sequence operations," *Signal Processing: Image Communication*, vol. 52, pp. 6 – 19, 2017.

[59] W. Zhang, K.-w. Wong, H. Yu, and Z.-l. Zhu, "An image encryption scheme using reverse 2-dimensional chaotic map and dependent diffusion," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 8, pp. 2066–2080, 2013.

[60] M. Ghebleh, A. Kanso, and H. Noura, "An image encryption scheme based on irregularly decimated chaotic maps," *Signal Processing: Image Communication*, vol. 29, no. 5, pp. 618–627, 2014.

[61] I. Mehra and N. K. Nishchal, "Optical asymmetric image encryption using gyrator wavelet transform," *Optics Communications*, vol. 354, pp. 344–352, 2015.

[62] N. Rawat, B. Kim, and R. Kumar, "Fast digital image encryption based on compressive sensing using structurally random matrices and arnold transform technique," *Optik - International Journal for Light and Electron Optics*, vol. 127, no. 4, pp. 2282 – 2286, 2016.

[63] Y. Zhang, B. Xu, and N. Zhou, "A novel image compression–encryption hybrid algorithm based on the analysis sparse representation," *Optics Communications*, vol. 392, pp. 223–233, 2017.

[64] M. Khan and T. Shah, "A novel statistical analysis of chaotic s-box in image encryption," *3D Research*, vol. 5, no. 3, pp. 1–8, 2014.

[65] A. A. A. El-Latif and X. Niu, "A hybrid chaotic system and cyclic elliptic curve for image encryption," *AEU-International Journal of Electronics and Communications*, vol. 67, no. 2, pp. 136–143, 2013.

[66] B. A. Forouzan and D. Mukhopadhyay, *Cryptography and Network Security (Sie)*. McGraw-Hill Education, 2011.

[67] M. Khan and Z. Asghar, "A novel construction of substitution box for image encryption applications with gingerbreadman chaotic map and s8 permutation," *Neural Computing and Applications*, vol. 29, no. 4, pp. 993–999, 20185.

[68] U. Cavusoglu, S. Kacar, I. Pehlivan, and A. Zengin, "Secure image encryption algorithm design using a novel chaos based s-box," *Chaos, Solitons & Fractals*, vol. 95, pp. 92–101, 2017.

[69] E. Chen, L. Min, and G. Chen, "Discrete chaotic systems with one-line equilibria and their application to image encryption," *International Journal of Bifurcation and Chaos*, vol. 27, no. 03, p. 1750046, 2017.

[70] R. Matthews, "On the derivation of a chaotic encryption algorithm," *Cryptologia*, vol. 13, no. 1, pp. 29–42, 1989.

[71] C. Li, G. Luo, K. Qin, and C. Li, "An image encryption scheme based on chaotic tent map," *Nonlinear Dynamics*, vol. 87, no. 1, pp. 127–133, 2017.

[72] M. Usama, M. K. Khan, K. Alghathbar, and C. Lee, "Chaos-based secure satellite imagery cryptosystem," *Computers & Mathematics with Applications*, vol. 60, no. 2, pp. 326–337, 2010.

[73] W. Wen, "Security analysis of a color image encryption scheme based on skew tent map and hyper chaotic system of 6th-order cnn against chosen-plaintext attack," *Multimedia Tools and Applications*, vol. 75, no. 6, pp. 3553–3560, 2016.

[74] Z.-l. Zhu, W. Zhang, K.-w. Wong, and H. Yu, "A chaos-based symmetric image encryption scheme using a bit-level permutation," *Information Sciences*, vol. 181, no. 6, pp. 1171–1186, 2011.

[75] C. Li, "Cracking a hierarchical chaotic image encryption algorithm based on permutation," *Signal Processing*, vol. 118, pp. 203–210, 2016.

[76] Y.-Q. Zhang and X.-Y. Wang, "Analysis and improvement of a chaos-based symmetric image encryption scheme using a bit-level permutation," *Nonlinear Dynamics*, vol. 77, no. 3, pp. 687–698, 2014.

[77] X. Zhang, G. Zhu, and S. Ma, "Remote-sensing image encryption in hybrid domains," *Optics Communications*, vol. 285, no. 7, pp. 1736–1743, 2012.

[78] X. Wang, Y. L. Yin, and H. Yu, "Finding collisions in the full sha-1," in *Crypto*, vol. 3621, pp. 17–36, Springer, 2005.

[79] X.-Y. Wang, Y.-Q. Zhang, and X.-M. Bao, "A novel chaotic image encryption scheme using dna sequence operations," *Optics and Lasers in Engineering*, vol. 73, pp. 53–61, 2015.

[80] W. Zhang, H. Yu, Y.-l. Zhao, and Z.-l. Zhu, "Image encryption based on three-dimensional bit matrix permutation," *Signal Processing*, vol. 118, pp. 36–50, 2016.

[81] J. Wu, X. Liao, and B. Yang, "Cryptanalysis and enhancements of image encryption based on three-dimensional bit matrix permutation," *Signal Processing*, vol. 142, pp. 292–300, 2018.

[82] M. Kumar, A. Iqbal, and P. Kumar, "A new rgb image encryption algorithm based on dna encoding and elliptic curve diffie–hellman cryptography," *Signal Processing*, vol. 125, pp. 187–202, 2016.

[83] A. Akhavan, A. Samsudin, and A. Akhshani, "Cryptanalysis of an image encryption algorithm based on dna encoding," *Optics & Laser Technology*, vol. 95, pp. 94–99, 2017.

[84] G. Zhou, D. Zhang, Y. Liu, Y. Yuan, and Q. Liu, "A novel image encryption algorithm based on chaos and line map," *Neurocomputing*, vol. 169, pp. 150–157, 2015.

[85] L. Chen, B. Ma, X. Zhao, and S. Wang, "Differential cryptanalysis of a novel image encryption algorithm based on chaos and line map," *Nonlinear Dynamics*, vol. 87, no. 3, pp. 1797–1807, 2017.

[86] B. Norouzi, S. Mirzakuchaki, S. M. Seyedzadeh, and M. R. Mosavi, "A simple, sensitive and secure image encryption algorithm based on hyper-chaotic system with only one round diffusion process," *Multimedia tools and applications*, vol. 71, no. 3, pp. 1469–1497, 2014.

[87] Z. Parvin, H. Seyedarabi, and M. Shamsi, "A new secure and sensitive image encryption scheme based on new substitution with chaotic function," *Multimedia Tools and Applications*, vol. 75, no. 17, pp. 10631–10648, 2016.

[88] S. Behnia, A. Akhshani, H. Mahmodi, and A. Akhavan, "A novel algorithm for image encryption based on mixture of chaotic maps," *Chaos, Solitons & Fractals*, vol. 35, no. 2, pp. 408–419, 2008.

[89] G. Ye, "Image scrambling encryption algorithm of pixel bit based on chaos map," *Pattern Recognition Letters*, vol. 31, no. 5, pp. 347–354, 2010.

[90] C. Li and D. Lin, "Cryptanalyzing an image encryption algorithm based on autoblocking and electrocardiography," *arXiv preprint arXiv:1711.01858*, 2017.

[91] O. Mirzaei, M. Yaghoobi, and H. Irani, "A new image encryption method: parallel sub-image encryption with hyper chaos," *Nonlinear Dynamics*, vol. 67, no. 1, pp. 557–566, 2012.

[92] A. Kanso and M. Ghebleh, "A novel image encryption algorithm based on a 3d chaotic map," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 7, pp. 2943–2959, 2012.

[93] X.-Y. Wang, T. Wang, D.-H. Xu, and F. Chen, "a selective image encryption based on couple spatial chaotic systems," *International Journal of Modern Physics B*, vol. 28, no. 06, p. 1450023, 2014.

[94] Y.-Q. Zhang and X.-Y. Wang, "A symmetric image encryption algorithm based on mixed linear–nonlinear coupled map lattice," *Information Sciences*, vol. 273, pp. 329–351, 2014.

[95] X. Wu, D. Wang, J. Kurths, and H. Kan, "A novel lossless color image encryption scheme using 2d dwt and 6d hyperchaotic system," *Information Sciences*, vol. 349, pp. 137–153, 2016.

[96] W. Liu, K. Sun, and C. Zhu, "A fast image encryption algorithm based on chaotic map," *Optics and Lasers in Engineering*, vol. 84, pp. 26–36, 2016.

[97] X. Wang, Q. Wang, and Y. Zhang, "A fast image algorithm based on rows and columns switch," *Nonlinear Dynamics*, vol. 79, no. 2, pp. 1141–1149, 2015.

[98] F. Ahmed, M. Siyal, and V. U. Abbas, "A perceptually scalable and jpeg compression tolerant image encryption scheme," in *Image and Video Technology (PSIVT), 2010 Fourth Pacific-Rim Symposium on*, pp. 232–238, IEEE, 2010.

[99] Z. Hua and Y. Zhou, "Design of image cipher using block-based scrambling and image filtering," *Information Sciences*, vol. 396, pp. 97–113, 2017.

[100] C. Pak and L. Huang, "A new color image encryption using combination of the 1d chaotic map," *Signal Processing*, vol. 138, pp. 129–137, 2017.

[101] Y. Li, C. Wang, and H. Chen, "A hyper-chaos-based image encryption algorithm using pixel-level permutation and bit-level permutation," *Optics and Lasers in Engineering*, vol. 90, pp. 238–246, 2017.

[102] Y. Wang, K.-W. Wong, X. Liao, and G. Chen, "A new chaos-based fast image encryption algorithm," *Applied soft computing*, vol. 11, no. 1, pp. 514–522, 2011.

[103] J.-x. Chen, Z.-l. Zhu, C. Fu, and H. Yu, "Optical image encryption scheme using 3-D chaotic map based joint image scrambling and random encoding in gyrator domains," *Optics Communications*, vol. 341, pp. 263–270, 2015.

[104] N. Zhou, S. Pan, S. Cheng, and Z. Zhou, "Image compression–encryption scheme based on hyper-chaotic system and 2d compressive sensing," *Optics & Laser Technology*, vol. 82, pp. 121–133, 2016.

[105] Q. Zhang, L. Liu, and X. Wei, "Improved algorithm for image encryption based on dna encoding and multi-chaotic maps," *AEU-International Journal of Electronics and Communications*, vol. 68, no. 3, pp. 186–192, 2014.

[106] X. Li, L. Wang, Y. Yan, and P. Liu, "An improvement color image encryption algorithm based on dna operations and real and complex chaotic systems," *Optik-International Journal for Light and Electron Optics*, vol. 127, no. 5, pp. 2558–2565, 2016.

[107] X. Wu, H. Kan, and J. Kurths, "A new color image encryption scheme based on dna sequences and multiple improved 1d chaotic maps," *Applied Soft Computing*, vol. 37, pp. 24–39, 2015.

[108] X. yuan Wang, H. li Zhang, and X. mei Bao, "Color image encryption scheme using cml and dna sequence operations," *Biosystems*, vol. 144, pp. 18 – 26, 2016.

[109] B. Mondal and T. Mandal, "A light weight secure image encryption scheme based on chaos & dna computing," *Journal of King Saud University - Computer and Information Sciences*, pp. –, 2016.

[110] X. Wang and D. Luan, "A novel image encryption algorithm using chaos and reversible cellular automata," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 11, pp. 3075–3085, 2013.

[111] X. Li, D. Xiao, and Q.-H. Wang, "Error-free holographic frames encryption with ca pixel-permutation encoding algorithm," *Optics and Lasers in Engineering*, vol. 100, pp. 200–207, 2018.

[112] X. Wang and D. Luan, "A novel image encryption algorithm using chaos and reversible cellular automata," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 11, pp. 3075 – 3085, 2013.

[113] A. Bakhshandeh and Z. Eslami, "An authenticated image encryption scheme based on chaotic maps and memory cellular automata," *Optics and Lasers in Engineering*, vol. 51, no. 6, pp. 665 – 673, 2013.

[114] P. Ping, F. Xu, and Z.-J. Wang, "Image encryption based on non-affine and balanced cellular automata," *Signal Processing*, vol. 105, pp. 419 – 429, 2014.

[115] X. W. Li, S. J. Cho, and S. T. Kim, "A 3d image encryption technique using computer-generated integral imaging and cellular automata transform," *Optik - International Journal for Light and Electron Optics*, vol. 125, no. 13, pp. 2983 – 2990, 2014.

[116] F. K. Mohamed, "A parallel block-based encryption schema for digital images using reversible cellular automata," *Engineering Science and Technology, an International Journal*, vol. 17, no. 2, pp. 85–94, 2014.

[117] R. Enayatifar, H. J. Sadaei, A. H. Abdullah, M. Lee, and I. F. Isnin, "A novel chaotic based image encryption using a hybrid model of deoxyribonucleic acid and cellular automata," *Optics and Lasers in Engineering*, vol. 71, pp. 33 – 41, 2015.

[118] Y.-G. Yang, J. Tian, H. Lei, Y.-H. Zhou, and W.-M. Shi, "Novel quantum image encryption using one-dimensional quantum cellular automata," *Information Sciences*, vol. 345, pp. 257–270, 2016.

[119] X. Li, C. Li, and I.-K. Lee, "Chaotic image encryption using pseudo-random masks and pixel mapping," *Signal Processing*, vol. 125, pp. 48–63, 2016.

[120] A. Y. Niyat, M. H. Moattar, and M. N. Torshiz, "Color image encryption based on hybrid hyper-chaotic system and cellular automata," *Optics and Lasers in Engineering*, vol. 90, pp. 225 – 237, 2017.

[121] N. Sreelaja and G. V. Pai, "Stream cipher for binary image encryption using ant colony optimization based key generation," *Applied Soft Computing*, vol. 12, no. 9, pp. 2879–2895, 2012.

[122] R. Enayatifar, A. H. Abdullah, and M. Lee, "A weighted discrete imperialist competitive algorithm (wdica) combined with chaotic map for image encryption," *Optics and Lasers in Engineering*, vol. 51, no. 9, pp. 1066–1077, 2013.

[123] N. A. Abbas, "Image encryption based on independent component analysis and arnolds cat map," *Egyptian Informatics Journal*, vol. 17, no. 1, pp. 139–146, 2016.

[124] K. M. Talarposhti and M. K. Jamei, "A secure image encryption method based on dynamic harmony search (DHS) combined with chaotic map," *Optics and Lasers in Engineering*, vol. 81, pp. 21–34, 2016.

[125] S. Behnia, A. Akhavan, A. Akhshani, and A. Samsudin, "Image encryption based on the jacobian elliptic maps," *Journal of Systems and Software*, vol. 86, no. 9, pp. 2429–2438, 2013.

[126] S. Nagaraj, G. Raju, and K. K. Rao, "Image encryption using elliptic curve cryptograhy and matrix," *Procedia Computer Science*, vol. 48, pp. 276–281, 2015.

[127] H. Liu, X. Wang, and A. Kadir, "Color image encryption using choquet fuzzy integral and hyper chaotic system," *Optik - International Journal for Light and Electron Optics*, vol. 124, no. 18, pp. 3527 – 3533, 2013.

[128] S. M. Seyedzadeh, B. Norouzi, and S. Mirzakuchaki, "Rgb color image encryption based on choquet fuzzy integral," *Journal of Systems and Software*, vol. 97, pp. 128 – 139, 2014.

[129] H. Chen, X. Du, Z. Liu, and C. Yang, "Color image encryption based on the affine transform and gyrator transform," *Optics and Lasers in Engineering*, vol. 51, no. 6, pp. 768–775, 2013.

[130] N. Singh and A. Sinha, "Gyrator transform-based optical image encryption, using chaos," *Optics and Lasers in Engineering*, vol. 47, no. 5, pp. 539–546, 2009.

[131] Q. Wang, Q. Guo, L. Lei, and J. Zhou, "Linear exchanging operation and random phase encoding in gyrator transform domain for double image encryption," *Optik-International Journal for Light and Electron Optics*, vol. 124, no. 24, pp. 6707–6712, 2013.

[132] Q. Wang, Q. Guo, and L. Lei, "Multiple-image encryption system using cascaded phase mask encoding and a modified gerchberg–saxton algorithm in gyrator domain," *Optics Communications*, vol. 320, pp. 12–21, 2014.

[133] M. R. Abuturab, "An asymmetric single-channel color image encryption based on hartley transform and gyrator transform," *Optics and Lasers in Engineering*, vol. 69, pp. 49–57, 2015.

[134] J.-x. Chen, Z.-l. Zhu, C. Fu, L.-b. Zhang, and H. Yu, "Analysis and improvement of a double-image encryption scheme using pixel scrambling technique in gyrator domains," *Optics and Lasers in Engineering*, vol. 66, pp. 1–9, 2015.

[135] L. Yao, C. Yuan, J. Qiang, S. Feng, and S. Nie, "An asymmetric color image encryption method by using deduced gyrator transform," *Optics and Lasers in Engineering*, vol. 89, pp. 72–79, 2017.

[136] Y.-Y. Wang, Y.-R. Wang, Y. Wang, H.-J. Li, and W.-J. Sun, "Optical image encryption based on binary fourier transform computer-generated hologram and pixel scrambling technology," *Optics and lasers in engineering*, vol. 45, no. 7, pp. 761–765, 2007.

[137] Q. Guo, Z. Liu, and S. Liu, "Color image encryption by using Arnold and discrete fractional random transforms in IHS space," *Optics and Lasers in Engineering*, vol. 48, no. 12, pp. 1174–1181, 2010.

[138] Y. Li, F. Zhang, Y. Li, and R. Tao, "Asymmetric multiple-image encryption based on the cascaded fractional fourier transform," *Optics and Lasers in Engineering*, vol. 72, pp. 18–25, 2015.

[139] X.-W. Li and I.-K. Lee, "Modified computational integral imaging-based double image encryption using fractional fourier transform," *Optics and Lasers in Engineering*, vol. 66, pp. 112–121, 2015.

[140] Q. Ran, L. Yuan, and T. Zhao, "Image encryption based on nonseparable fractional fourier transform and chaotic map," *Optics Communications*, vol. 348, pp. 43–49, 2015.

[141] H. Zhao, J. Liu, J. Jia, N. Zhu, J. Xie, and Y. Wang, "Multiple-image encryption based on position multiplexing of fresnel phase," *Optics Communications*, vol. 286, pp. 85–90, 2013.

[142] Q. Wang, Q. Guo, L. Lei, and J. Zhou, "Single-beam image encryption using spatially separated ciphertexts based on interference principle in the fresnel domain," *Optics Communications*, vol. 333, pp. 151–158, 2014.

[143] Y. Wang, C. Quan, and C. Tay, "Nonlinear multiple-image encryption based on mixture retrieval algorithm in fresnel domain," *Optics Communications*, vol. 330, pp. 91–98, 2014.

[144] Y. Wang, C. Quan, and C. Tay, "Optical color image encryption without information disclosure using phase-truncated fresnel transform and a random amplitude mask," *Optics Communications*, vol. 344, pp. 147–155, 2015.

[145] Y. Luo, M. Du, and J. Liu, "A symmetrical image encryption scheme in wavelet and time domain," *Communications in Nonlinear Science and Numerical Simulation*, vol. 20, no. 2, pp. 447–460, 2015.

[146] A. Kanso and M. Ghebleh, "An algorithm for encryption of secret images into meaningful images," *Optics and Lasers in Engineering*, vol. 90, pp. 196–208, 2017.

[147] J. B. Lima, F. Madeiro, and F. Sales, "Encryption of medical images based on the cosine number transform," *Signal Processing: Image Communication*, vol. 35, pp. 1–8, 2015.

[148] J. Wu, F. Guo, Y. Liang, and N. Zhou, "Triple color images encryption algorithm based on scrambling and the reality-preserving fractional discrete cosine transform," *Optik-International Journal for Light and Electron Optics*, vol. 125, no. 16, pp. 4474–4479, 2014.

[149] L. Yaru and W. Jianhua, "New image encryption combining fractional dct via polynomial interpolation with dependent scrambling and diffusion," *The Journal of China Universities of Posts and Telecommunications*, vol. 22, no. 5, pp. 1–9, 2015.

[150] X. Wang and G. He, "Cryptanalysis on a novel image encryption method based on total shuffling scheme," *Optics Communications*, vol. 284, no. 24, pp. 5804–5807, 2011.

[151] G. Zhang and Q. Liu, "A novel image encryption method based on total shuffling scheme," *Optics Communications*, vol. 284, no. 12, pp. 2775–2780, 2011.

[152] M. Murillo-Escobar, C. Cruz-Hernández, F. Abundiz-Pérez, R. López-Gutiérrez, and O. A. Del Campo, "A rgb image encryption algorithm based on total plain image characteristics and chaos," *Signal Processing*, vol. 109, pp. 119–131, 2015.

[153] H. Fan, M. Li, D. Liu, and K. An, "Cryptanalysis of a plaintext-related chaotic rgb image encryption scheme using total plain image characteristics," *Multimedia Tools and Applications*, pp. 1–25, 2017.

[154] L. Zeng, R. Liu, L. Y. Zhang, Y. Liu, and K.-W. Wong, "Cryptanalyzing an image encryption algorithm based on scrambling and veginere cipher," *Multimedia Tools and Applications*, vol. 75, no. 10, pp. 5439–5453, 2016.

[155] S. Li, Y. Zhao, B. Qu, *et al.*, "Image scrambling based on chaotic sequences and veginère cipher," *Multimedia tools and applications*, vol. 66, no. 3, pp. 573–588, 2013.

[156] X. Su, W. Li, and H. Hu, "Cryptanalysis of a chaos-based image encryption scheme combining dna coding and entropy," *Multimedia Tools and Applications*, vol. 76, no. 12, pp. 14021–14033, 2017.

[157] P. Zhen, G. Zhao, L. Min, and X. Jin, "Chaos-based image encryption scheme combining dna coding and entropy," *Multimedia Tools and Applications*, vol. 75, no. 11, pp. 6303–6319, 2016.

[158] X. Zhang, W. Nie, Y. Ma, and Q. Tian, "Cryptanalysis and improvement of an image encryption algorithm based on hyper-chaotic system and dynamic s-box," *Multimedia Tools and Applications*, vol. 76, no. 14, pp. 15641–15659, 2017.

[159] Y. Liu, X. Tong, and J. Ma, "Image encryption algorithm based on hyper-chaotic system and dynamic s-box," *Multimedia Tools and Applications*, vol. 75, no. 13, pp. 7739–7759, 2016.

[160] B. Norouzi and S. Mirzakuchaki, "Breaking a novel image encryption scheme based on an improper fractional order chaotic system," *Multimedia Tools and Applications*, vol. 76, no. 2, pp. 1817–1826, 2017.

[161] J. Zhao, S. Wang, Y. Chang, and X. Li, "A novel image encryption scheme based on an improper fractional-order chaotic system," *Nonlinear Dynamics*, vol. 80, no. 4, pp. 1721–1729, 2015.

[162] A. L. Da Cunha, J. Zhou, and M. N. Do, "The nonsubsampled contourlet transform: theory, design, and applications," *IEEE transactions on image processing*, vol. 15, no. 10, pp. 3089–3101, 2006.

[163] X. Xie, J. Lai, and W.-S. Zheng, "Extraction of illumination invariant facial features from a single image using nonsubsampled contourlet transform," *Pattern Recognition*, vol. 43, no. 12, pp. 4177–4189, 2010.

[164] E. Chen, J. Wang, L. Qi, and W. Lv, "A novel multiscale edge detection approach based on nonsubsampled contourlet transform and edge tracking," *Mathematical Problems in Engineering*, vol. 2015, 2015.

[165] A. Shah and S. Gupta, "Optimum multiscale decomposition in nsct-based single image super resolution," *The Imaging Science Journal*, vol. 64, no. 3, pp. 140–151, 2016.

[166] X. Wang and C. Chen, "Image fusion for synthetic aperture radar and multispectral images based on sub-band-modulated non-subsampled contourlet transform and pulse coupled neural network methods," *The Imaging Science Journal*, vol. 64, no. 2, pp. 87–93, 2016.

[167] "The USC-SIPI image database," *Signal and Image Processing Institute*, http://sipi.usc.edu/database/ 2017.

[168] A. Kulsoom, D. Xiao, S. A. Abbas, *et al.*, "An efficient and noise resistive selective image encryption scheme for gray images based on chaotic maps and dna complementary rules," *Multimedia Tools and Applications*, vol. 75, no. 1, pp. 1–23, 2016.

[169] Y. Wu, J. P. Noonan, and S. Agaian, "Npcr and uaci randomness tests for image encryption," *Cyber journals: multidisciplinary journals in science and technology, Journal of Selected Areas in Telecommunications (JSAT)*, vol. 1, no. 2, pp. 31–38, 2011.

[170] R. Becheikh, T. Omrani, R. Rhouma, and S. Belghith, "Risc: a robust image symmetric cryptosystem," *Multimedia Tools and Applications*, vol. 77, no. 19, pp. 24615–24642, 2018.

[171] P. P. Mar and K. M. Latt, "New analysis methods on strict avalanche criterion of s-boxes," *World Academy of Science, Engineering and Technology*, vol. 48, no. 150-154, p. 25, 2008.

[172] S. Farwa, N. Muhammad, T. Shah, and S. Ahmad, "A novel image encryption based on algebraic s-box and arnold transform," *3D Research*, vol. 8, no. 3, p. 26, 2017.

[173] J. Zou and T. Weng, "A new image encryption instant communication method based on matrix transformation," in *Advances in Intelligent Information Hiding and Multimedia Signal Processing: Proceeding of the Twelfth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Nov., 21-23, 2016, Kaohsiung, Taiwan, Volume 1*, pp. 321–329, Springer, 2017.

[174] A. Vaish and M. Kumar, "Color image encryption using msvd, dwt and arnold transform in fractional fourier domain," *Optik-International Journal for Light and Electron Optics*, 2017.

[175] X. Zhang, X. Wang, and Y. Cheng, "Image encryption based on a genetic algorithm and a chaotic system," *IEICE Transactions on Communications*, vol. 98, no. 5, pp. 824–833, 2015.

[176] J. Ahmad and S. O. Hwang, "A secure image encryption scheme based on chaotic maps and affine transformation," *Multimedia Tools and Applications*, vol. 75, no. 21, pp. 13951–13976, 2016.

[177] C. E. Shannon, "Communication theory of secrecy systems," *Bell Labs Technical Journal*, vol. 28, no. 4, pp. 656–715, 1949.

[178] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[179] Q. Lin and J. Chen, "A novel micro-population immune multiobjective optimization algorithm," *Computers & Operations Research*, vol. 40, no. 6, pp. 1590–1601, 2013.

[180] T. Chen, M. Zhang, J. Wu, C. Yuen, and Y. Tong, "Image encryption and compression based on kronecker compressed sensing and elementary cellular automata scrambling," *Optics & Laser Technology*, vol. 84, pp. 118–133, 2016.

[181] R. Kumar and B. Bhaduri, "Optical image encryption using kronecker product and hybrid phase masks," *Optics & Laser Technology*, vol. 95, pp. 51–55, 2017.