

Lightweight image encryption algorithms: design and evaluation

by

Yousef Alghamdi

B.S., King Abdulaziz University, 2012

M.S., North Carolina Agricultural and Technical State University, 2015

---

AN ABSTRACT OF A DISSERTATION

submitted in partial fulfillment of the  
requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Computer Science  
Carl R. Ice College of Engineering

KANSAS STATE UNIVERSITY  
Manhattan, Kansas

2023

# Abstract

In an era dominated by increasing use of multimedia data such as images and videos, ensuring the security and confidentiality of images with real-time encryption is of greatest importance. Traditional encryption algorithms are secure, widely used, and recommended, yet they are not suitable nor computationally efficient for encrypting multimedia data due to the large size and high redundancy inherent in multimedia data. Thus, specialized algorithms for multimedia data encryption are needed. This dissertation explores lightweight image encryption algorithms, specifically designed to address time and resource constraints of real-time image encryption while maintaining the confidentiality and integrity of the multimedia data.

The dissertation classifies image encryption based on the techniques used into seven different approaches and analyzes the strengths and weaknesses of each approach. It subsequently introduces and evaluates three novel algorithms designed to encrypt images with low complexity, high efficiency, and reliable security. These algorithms rely on a combination of permutation, substitution, and pseudorandom keystreams to ensure the security of the encrypted images. The first algorithm is based on chaotic systems. The algorithm is implemented using logistic map, permutations, AES S-box, and a plaintext related SHA-2 hash. The second algorithm is based on Trivium cipher. the algorithm is implemented to work on multi-rounds of encryption using pixel-based row and column permutations, and bit-level substitution. For the third algorithm, the Ascon algorithm selected by the National Institute of Standards and Technology (NIST) to standardize lightweight cryptography applications is evaluated for image encryption.

To evaluate the proposed algorithms, a comprehensive set of security, quality, and efficiency valuation metrics is utilized to assess the proposed algorithms and compare them to contemporary image encryption algorithms.



Lightweight image encryption algorithms: design and evaluation

by

Yousef Alghamdi

B.S., King Abdulaziz University, 2012

M.S., North Carolina Agricultural and Technical State University, 2015

---

A DISSERTATION

submitted in partial fulfillment of the  
requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Computer Science  
Carl R. Ice College of Engineering

KANSAS STATE UNIVERSITY  
Manhattan, Kansas

2023

Approved by:

Major Professor  
Arslan Munir

# Copyright

© Yousef Alghamdi 2023.

# Abstract

In an era dominated by increasing use of multimedia data such as images and videos, ensuring the security and confidentiality of images with real-time encryption is of greatest importance. Traditional encryption algorithms are secure, widely used, and recommended, yet they are not suitable nor computationally efficient for encrypting multimedia data due to the large size and high redundancy inherent in multimedia data. Thus, specialized algorithms for multimedia data encryption are needed. This dissertation explores lightweight image encryption algorithms, specifically designed to address time and resource constraints of real-time image encryption while maintaining the confidentiality and integrity of the multimedia data.

The dissertation classifies image encryption based on the techniques used into seven different approaches and analyzes the strengths and weaknesses of each approach. It subsequently introduces and evaluates three novel algorithms designed to encrypt images with low complexity, high efficiency, and reliable security. These algorithms rely on a combination of permutation, substitution, and pseudorandom keystreams to ensure the security of the encrypted images. The first algorithm is based on chaotic systems. The algorithm is implemented using logistic map, permutations, AES S-box, and a plaintext related SHA-2 hash. The second algorithm is based on Trivium cipher. the algorithm is implemented to work on multi-rounds of encryption using pixel-based row and column permutations, and bit-level substitution. For the third algorithm, the Ascon algorithm selected by the National Institute of Standards and Technology (NIST) to standardize lightweight cryptography applications is evaluated for image encryption.

To evaluate the proposed algorithms, a comprehensive set of security, quality, and efficiency valuation metrics is utilized to assess the proposed algorithms and compare them to contemporary image encryption algorithms.

# Table of Contents

|                                                                                  |      |
|----------------------------------------------------------------------------------|------|
| List of Figures . . . . .                                                        | x    |
| List of Tables . . . . .                                                         | xiii |
| Acknowledgements . . . . .                                                       | xix  |
| Dedication . . . . .                                                             | xx   |
| 1 Image Encryption Algorithms: A Survey of Design and Evaluation Metrics . . . . | 1    |
| 1.1 Introduction . . . . .                                                       | 1    |
| 1.2 Image Encryption Algorithms . . . . .                                        | 2    |
| 1.2.1 Traditional Ciphers . . . . .                                              | 3    |
| 1.2.2 Chaotic Systems . . . . .                                                  | 5    |
| 1.2.3 DNA Encoding . . . . .                                                     | 13   |
| 1.2.4 Neural Networks . . . . .                                                  | 15   |
| 1.2.5 Frequency Domain . . . . .                                                 | 16   |
| 1.2.6 Compressive Sensing . . . . .                                              | 17   |
| 1.2.7 Meaningful Encryption . . . . .                                            | 19   |
| 1.3 Evaluation Metrics . . . . .                                                 | 20   |
| 1.3.1 Correlation Coefficient Analysis . . . . .                                 | 21   |
| 1.3.2 Histogram Analysis . . . . .                                               | 22   |
| 1.3.3 Entropy Analysis . . . . .                                                 | 28   |
| 1.3.4 Gray-Level Co-Occurrence Matrix (GLCM) Analysis . . . . .                  | 29   |
| 1.3.5 Encryption Quality . . . . .                                               | 30   |

|        |                                                     |    |
|--------|-----------------------------------------------------|----|
| 1.3.6  | Resistance Against Differential Attacks . . . . .   | 33 |
| 1.3.7  | Resistance to Noise and Data Loss . . . . .         | 35 |
| 1.3.8  | Key Space and Key Sensitivity . . . . .             | 36 |
| 1.3.9  | NIST SP 800-22 Test Suite . . . . .                 | 37 |
| 1.4    | Discussion . . . . .                                | 37 |
| 1.5    | Conclusion . . . . .                                | 40 |
| 2      | Image Encryption Based on Chaos . . . . .           | 42 |
| 2.1    | Introduction . . . . .                              | 42 |
| 2.2    | Proposed Image Encryption Algorithm . . . . .       | 44 |
| 2.3    | Results and Analysis . . . . .                      | 49 |
| 2.3.1  | Encryption Speed . . . . .                          | 50 |
| 2.3.2  | Correlation Coefficient Analysis . . . . .          | 52 |
| 2.3.3  | Histogram Analysis . . . . .                        | 54 |
| 2.3.4  | Entropy Analysis . . . . .                          | 55 |
| 2.3.5  | Encryption Quality . . . . .                        | 55 |
| 2.3.6  | Resistance to Differential Attacks . . . . .        | 60 |
| 2.3.7  | Resistance to Noise and Data Loss Attacks . . . . . | 62 |
| 2.3.8  | Homogeneity . . . . .                               | 63 |
| 2.3.9  | Contrast . . . . .                                  | 63 |
| 2.3.10 | Energy . . . . .                                    | 64 |
| 2.3.11 | Key Space and Sensitivity Analyses . . . . .        | 65 |
| 2.3.12 | NIST SP 800-22 Test . . . . .                       | 67 |
| 2.4    | Conclusion . . . . .                                | 67 |
| 3      | Image Encryption Based on Trivium . . . . .         | 70 |
| 3.1    | Introduction . . . . .                              | 70 |
| 3.2    | Preliminaries . . . . .                             | 71 |

|       |                                                                     |     |
|-------|---------------------------------------------------------------------|-----|
| 3.2.1 | Nonlinear-Feedback Shift Registers (NLFSR) . . . . .                | 72  |
| 3.2.2 | Trivium Cipher . . . . .                                            | 72  |
| 3.3   | Proposed Image Encryption Algorithm . . . . .                       | 73  |
| 3.4   | Security of The Proposed Algorithm . . . . .                        | 77  |
| 3.4.1 | Key Space . . . . .                                                 | 77  |
| 3.4.2 | Rounds and Key Schedule . . . . .                                   | 77  |
| 3.4.3 | Pixel-Based Permutations . . . . .                                  | 78  |
| 3.4.4 | Bitwise XOR of Image Bits . . . . .                                 | 78  |
| 3.4.5 | Overall Security Level of the Proposed Image Encryption Algorithm . | 78  |
| 3.5   | Results and Analysis . . . . .                                      | 79  |
| 3.5.1 | Correlation Coefficient Analysis . . . . .                          | 79  |
| 3.5.2 | Histogram Analysis . . . . .                                        | 82  |
| 3.5.3 | Entropy Analysis . . . . .                                          | 85  |
| 3.5.4 | Gray-Level Co-Occurrence Matrix (GLCM) Analysis . . . . .           | 86  |
| 3.5.5 | Encryption Quality . . . . .                                        | 88  |
| 3.5.6 | Resistance Against Differential Attacks . . . . .                   | 91  |
| 3.5.7 | Resistance to Noise and Data Loss . . . . .                         | 94  |
| 3.5.8 | Key Sensitivity . . . . .                                           | 96  |
| 3.5.9 | NIST SP 800-22 Test . . . . .                                       | 98  |
| 3.6   | Conclusions . . . . .                                               | 98  |
| 4     | Evaluating The Ascon128 for Image Encryption . . . . .              | 101 |
| 4.1   | Introduction . . . . .                                              | 101 |
| 4.2   | Ascon Cipher . . . . .                                              | 102 |
| 4.2.1 | Ascon's Authenticated Encryption Mode . . . . .                     | 102 |
| 4.2.2 | Ascon's Permutation . . . . .                                       | 105 |
| 4.3   | Results and Analysis . . . . .                                      | 106 |
| 4.3.1 | Correlation Coefficient Analysis . . . . .                          | 107 |

|       |                                                           |     |
|-------|-----------------------------------------------------------|-----|
| 4.3.2 | Histogram Analysis . . . . .                              | 110 |
| 4.3.3 | Entropy Analysis . . . . .                                | 114 |
| 4.3.4 | Gray-Level Co-Occurrence Matrix (GLCM) Analysis . . . . . | 114 |
| 4.3.5 | Encryption Quality . . . . .                              | 117 |
| 4.3.6 | Resistance Against Differential Attacks . . . . .         | 119 |
| 4.3.7 | Resistance to Noise and Data Loss . . . . .               | 121 |
| 4.3.8 | Key Sensitivity . . . . .                                 | 123 |
| 4.3.9 | NIST SP 800-22 Test . . . . .                             | 124 |
| 4.4   | Conclusion . . . . .                                      | 124 |
| 5     | Conclusion . . . . .                                      | 126 |
|       | Bibliography . . . . .                                    | 129 |

# List of Figures

|     |                                                                                                                                                                                                                                 |    |
|-----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 1.1 | Different approaches to image encryption based on used techniques. . . . .                                                                                                                                                      | 3  |
| 1.2 | Logistic map bifurcation diagram. . . . .                                                                                                                                                                                       | 6  |
| 1.3 | Tent map bifurcation diagram. . . . .                                                                                                                                                                                           | 9  |
| 1.4 | Henon map bifurcation diagram. . . . .                                                                                                                                                                                          | 10 |
| 1.5 | Most commonly used evaluation metrics. . . . .                                                                                                                                                                                  | 21 |
| 2.1 | Proposed image encryption algorithm's flowchart. . . . .                                                                                                                                                                        | 45 |
| 2.2 | Logistic map bifurcation diagram. . . . .                                                                                                                                                                                       | 46 |
| 2.3 | The proposed algorithm applied to a $4 \times 4$ sample. . . . .                                                                                                                                                                | 49 |
| 2.4 | Images used to test the proposed algorithm: <b>(a)</b> Baboon, <b>(b)</b> Cameraman,<br><b>(c)</b> Clock, <b>(d)</b> Male, <b>(e)</b> Peppers, <b>(f)</b> Sailboat. . . . .                                                     | 50 |
| 2.5 | Sample images and their respective encrypted ciphers: (a) Baboon, (b) Male,<br>(c) Sailboat, (d) Baboon cipher, (e) Male cipher, (f) Sailboat cipher. . . . .                                                                   | 52 |
| 2.6 | Histograms of images and their respective ciphers: (a) Baboon histogram, (b)<br>Sailboat histogram, (c) Peppers histogram, (d) Baboon cipher histogram, (e)<br>Sailboat cipher histogram, (f) Peppers cipher histogram. . . . . | 55 |
| 2.7 | Avalanche effect illustrated in two images encrypted from the same plaintext<br>image with a change of 1-bit, (d) Peppers, (e) Peppers cipher, (f) Peppers<br>cipher where plaintext peppers have 1-bit change. . . . .         | 60 |
| 2.8 | Results of data loss (occlusion) attacks on the color image Baboon ( $512 \times 512$ ):<br>(a)-(c) different severities of occlusion attacks, (d)-(f) corresponding decrypted<br>images. . . . .                               | 63 |



|      |                                                                                                                                                                                                                                                                                                                                                                                                         |    |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 2.9  | Results of salt and pepper noise attack on the color image Baboon ( $512 \times 512$ ):<br>(a)-(c) different noise densities introduced, (d)-(f) corresponding decrypted<br>images. . . . .                                                                                                                                                                                                             | 64 |
| 2.10 | Results of key sensitivity tests on the color image Sailboat ( $512 \times 512$ ): (a)<br>plaintext image, (b) $C_1$ encrypted using key $K_1$ , (c) $C_2$ encrypted using key<br>$K_2$ , (d) difference of images: $-(b)-(c)-$ (e) $C_1$ decrypted using key $K_1$ , (f)<br>$C_1$ decrypted using key $K_2$ , (g) $C_2$ decrypted using key $K_1$ , (h) difference of<br>images: $-(f)-(g)-$ . . . . . | 67 |
| 3.1  | Trivium cipher flowchart <sup>1</sup> . . . . .                                                                                                                                                                                                                                                                                                                                                         | 73 |
| 3.2  | Proposed image encryption algorithm's flowchart. . . . .                                                                                                                                                                                                                                                                                                                                                | 75 |
| 3.3  | Sample images and their respective encrypted ciphers: <b>(a)</b> Clock, <b>(b)</b> Sail-<br>boat, <b>(c)</b> Male, <b>(d)</b> Clock cipher, <b>(e)</b> Sailboat cipher, <b>(f)</b> Male cipher. . . .                                                                                                                                                                                                   | 80 |
| 3.4  | Histograms of images and their respective ciphers: <b>(a)</b> Baboon histogram, <b>(b)</b><br>Cameraman histogram, <b>(c)</b> Peppers histogram, <b>(d)</b> Baboon cipher histogram,<br><b>(e)</b> Cameraman cipher histogram, <b>(f)</b> Peppers cipher histogram. . . . .                                                                                                                             | 83 |
| 3.5  | Avalanche effect on two cipher images encrypted from the same plaintext<br>images differing in only a single pixel, (a) Baboon, (b) Baboon cipher, (c)<br>Baboon cipher with 1 pixel changed in the Baboon image. . . . .                                                                                                                                                                               | 93 |
| 3.6  | Results of salt and pepper noise attack on the color image Peppers ( $512 \times 512$ ):<br><b>(a-c)</b> different noise densities introduced, <b>(d-f)</b> corresponding decrypted images. . . . .                                                                                                                                                                                                     | 96 |
| 3.7  | Results of occlusion attacks (data loss) on the color image Peppers ( $512 \times$<br>$512$ ): <b>(a-c)</b> different severities of occlusion attacks, <b>(d-f)</b> corresponding de-<br>crypted images. . . . .                                                                                                                                                                                        | 97 |

|     |                                                                                                                                                                                                                                                                                                                                                                                                                       |     |
|-----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 3.8 | Results of key sensitivity tests on the color image Sailboat ( $512 \times 512$ ): (a) plaintext image, (b) $C_1$ encrypted using key $k_1$ , (c) $C_2$ encrypted using key $k_2$ , (d) difference of images: $-(b) - (c)-$ , (e) $C_1$ decrypted using key $k_1$ , (f) $C_1$ decrypted using key $k_2$ , (g) $C_2$ decrypted using key $k_1$ , (h) difference of images: $-(f) - (g)-$ . . . . .                     | 98  |
| 4.1 | The mode of operation for Ascon authenticated encryption <sup>2</sup> . . . . .                                                                                                                                                                                                                                                                                                                                       | 103 |
| 4.2 | Images used to test the Ascon algorithm (a) Baboon, (b) Cameraman, (c) Clock, (d) Male, (e) Peppers, and (f) Sailboat. . . . .                                                                                                                                                                                                                                                                                        | 108 |
| 4.3 | Sample images and their respective encrypted ciphers: (a) Male, (b) Peppers, (c) Sailboat, (d) Male cipher, (e) Peppers cipher, (f) Sailboat cipher. . . .                                                                                                                                                                                                                                                            | 109 |
| 4.4 | Histograms of images and their respective ciphers: (a) Sailboat histogram, (b) Male histogram, (c) Sailboat cipher histogram, (d) Male cipher histogram. . . . .                                                                                                                                                                                                                                                      | 111 |
| 4.5 | Results of occlusion attacks on the color image Baboon ( $512 \times 512$ ): (a–c) different severities of occlusion attacks, (d–f) corresponding decrypted images. . . . .                                                                                                                                                                                                                                           | 122 |
| 4.6 | Results of salt and pepper noise attack on the color image Baboon ( $512 \times 512$ ): (a) salt and pepper noise with a density of 0.001, (b) corresponding decrypted image indicating that the image decryption failed. . . . .                                                                                                                                                                                     | 123 |
| 4.7 | Results of key sensitivity tests on the color image Sailboat ( $512 \times 512$ ): (a) plaintext image, (b) $C_1$ encrypted using key $K_1$ , (c) $C_2$ encrypted using key $K_2$ , (d) difference of encrypted images: $-(b) - (c)-$ , (e) $C_1$ decrypted using key $K_1$ , (f) $C_1$ decrypted using key $K_2$ , (g) $C_2$ decrypted using key $K_1$ , (h) difference of decrypted images: $-(f) - (g)-$ . . . . . | 124 |

# List of Tables

|     |                                                                                                                                                                        |    |
|-----|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 1.1 | Sample of different DNA coding rules. . . . .                                                                                                                          | 13 |
| 1.2 | DNA addition for DNA rule 1. . . . .                                                                                                                                   | 13 |
| 1.3 | Upper limit of Chi-square $\chi^2$ for different sized images. . . . .                                                                                                 | 24 |
| 1.4 | Upper limit of maximum deviation for different sized images. . . . .                                                                                                   | 25 |
| 1.5 | Upper limit of irregular deviation for different sized images. . . . .                                                                                                 | 26 |
| 1.6 | Strengths and weaknesses of different image encryption approaches. . . . .                                                                                             | 40 |
| 2.1 | An example of the generation of Matrix 1 ( $M_1$ ) from the logistic map. . . . .                                                                                      | 46 |
| 2.2 | Average encryption time (in seconds) of the proposed algorithm for different images dimensions. . . . .                                                                | 51 |
| 2.3 | Average encryption time (in seconds) of several color and gray scale $512 \times 512$ images for the proposed algorithm compared to different related methods. . . . . | 51 |
| 2.4 | Encryption time (in seconds) for the proposed algorithm compared to different related methods scaled to 3.5GHz CPU. . . . .                                            | 51 |
| 2.5 | Vertical, horizontal, and diagonal correlation coefficients of adjacent pixels of encrypted images. . . . .                                                            | 53 |
| 2.6 | Vertical, horizontal, and diagonal correlation coefficients of encrypted Baboon image of the proposed algorithm compared to different encryption methods. . . . .      | 53 |
| 2.7 | Vertical, horizontal, and diagonal correlation coefficients of plaintext and cipher images. . . . .                                                                    | 54 |
| 2.8 | Chi-square test values for plaintext images and cipher images encrypted by our proposed algorithm. . . . .                                                             | 54 |

|      |                                                                                                                                   |    |
|------|-----------------------------------------------------------------------------------------------------------------------------------|----|
| 2.9  | Global and local Shannon entropy values of images encrypted by our proposed algorithm. . . . .                                    | 56 |
| 2.10 | Global entropy of cipher image Baboon of the proposed algorithm compared with different encryption algorithms. . . . .            | 56 |
| 2.11 | MSE values of images encrypted by the proposed algorithm. . . . .                                                                 | 56 |
| 2.12 | Mean Square Error (MSE) for images encrypted by the proposed algorithm compared to related methods. . . . .                       | 57 |
| 2.13 | MAE values of images encrypted by the proposed algorithm. . . . .                                                                 | 57 |
| 2.14 | Average MAE values for images encrypted by the proposed algorithm compared to related methods. . . . .                            | 57 |
| 2.15 | Peak signal-to-noise ratio (PSNR) of images encrypted by the proposed algorithm. . . . .                                          | 58 |
| 2.16 | Peak signal-to-noise ratio (PSNR) of the images encrypted by the proposed algorithm compared to other encryption methods. . . . . | 58 |
| 2.17 | Maximum deviation results for proposed algorithm compared to related methods. . . . .                                             | 58 |
| 2.18 | Irregular deviation results for the proposed algorithm compared to other encryption methods. . . . .                              | 59 |
| 2.19 | Deviation from uniform histogram for the proposed algorithm compared with other encryption methods. . . . .                       | 59 |
| 2.20 | Avalanche effect $MSE_{av}$ , NPCR, and UACI values of images encrypted by the proposed algorithm. . . . .                        | 61 |
| 2.21 | NPCR and UACI of the proposed algorithm compared with other encryption methods. . . . .                                           | 62 |
| 2.22 | Homogeneity analysis for plaintext and cipher images. . . . .                                                                     | 65 |
| 2.23 | Contrast analysis for plaintext and cipher images. . . . .                                                                        | 65 |
| 2.24 | Energy analysis for plaintext and cipher images. . . . .                                                                          | 65 |

|      |                                                                                                                                                                        |    |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 2.25 | Comparison of homogeneity, contrast, and energy analysis of cipher image of the proposed algorithm and related methods. . . . .                                        | 66 |
| 2.26 | Results of NIST SP 800-22 tests for the encrypted Baboon image. . . . .                                                                                                | 68 |
| 3.1  | An example of the generation of Matrix <sub><i>i</i></sub> ( $M_i$ ) from the round key-stream. . .                                                                    | 75 |
| 3.2  | Vertical, horizontal, and diagonal correlation coefficients of adjacent pixels of encrypted images. . . . .                                                            | 81 |
| 3.3  | Vertical, horizontal, and diagonal correlation coefficients of the encrypted Baboon image for the proposed algorithm compared to different encryption methods. . . . . | 81 |
| 3.4  | Vertical, horizontal, and diagonal correlation coefficients of plaintext and cipher images. . . . .                                                                    | 82 |
| 3.5  | Chi-Square test values for cipher images produced by our proposed algorithm.                                                                                           | 84 |
| 3.6  | Chi-Square test values of the proposed algorithm compared to different encryption algorithms. . . . .                                                                  | 84 |
| 3.7  | Maximum deviation results of the proposed algorithm as compared to the other algorithms. . . . .                                                                       | 84 |
| 3.8  | Irregular deviation results of the proposed algorithm as compared to the other encryption algorithms. . . . .                                                          | 85 |
| 3.9  | Deviation from uniform histogram of the proposed algorithm as compared to other encryption algorithms. . . . .                                                         | 85 |
| 3.10 | Global and local entropy values of images encrypted by the proposed algorithm.                                                                                         | 86 |
| 3.11 | Global entropy of cipher image Baboon produced by the proposed algorithm compared to other encryption algorithms. . . . .                                              | 86 |
| 3.12 | Homogeneity analysis for plaintext and cipher images produced by the proposed encryption algorithm. . . . .                                                            | 87 |
| 3.13 | Contrast analysis for plaintext and cipher images produced by the proposed encryption algorithm. . . . .                                                               | 87 |

|      |                                                                                                                                                     |     |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 3.14 | Energy analysis for plaintext and cipher images produced by the proposed algorithm. . . . .                                                         | 88  |
| 3.15 | Comparison of homogeneity, contrast, and energy values of cipher images produced by the proposed algorithm against other encryption algorithms. . . | 88  |
| 3.16 | MSE values of images encrypted by the proposed algorithm. . . . .                                                                                   | 89  |
| 3.17 | MSE values for images encrypted by the proposed algorithm compared to related methods. . . . .                                                      | 89  |
| 3.18 | MAE values of images encrypted by the proposed algorithm. . . . .                                                                                   | 90  |
| 3.19 | MAE values for images encrypted by the proposed algorithm compared to related methods. . . . .                                                      | 90  |
| 3.20 | PSNR results of images encrypted by the proposed algorithm. . . . .                                                                                 | 91  |
| 3.21 | PSNR of the images encrypted by the proposed algorithm compared to other encryption algorithms. . . . .                                             | 91  |
| 3.22 | Avalanche effect $MSE_{av}$ values of images encrypted by the proposed algorithm.                                                                   | 92  |
| 3.23 | Avalanche effect $MSE_{av}$ values of the image Baboon encrypted by the proposed algorithm. . . . .                                                 | 92  |
| 3.24 | NPCR values of images encrypted by the proposed algorithm. . . . .                                                                                  | 94  |
| 3.25 | NPCR values of the image Cameraman encrypted by the proposed algorithm.                                                                             | 94  |
| 3.26 | NPCR of the proposed algorithm compared with other encryption methods.                                                                              | 94  |
| 3.27 | UACI values of images encrypted by the proposed algorithm. . . . .                                                                                  | 95  |
| 3.28 | UACI values of the image Clock encrypted by the proposed algorithm. . . .                                                                           | 95  |
| 3.29 | UACI of the proposed algorithm compared with other encryption methods. .                                                                            | 95  |
| 3.30 | Results of NIST SP 800-22 tests for the encrypted Baboon image. . . . .                                                                             | 99  |
| 4.1  | Recommended parameters for Ascon authenticated encryption <sup>2</sup> . . . . .                                                                    | 103 |
| 4.2  | The round-specific constants for each round of $p^a$ and $p^b$ <sup>2</sup> . . . . .                                                               | 106 |
| 4.3  | Ascon's S-box lookup table <sup>2</sup> . . . . .                                                                                                   | 106 |

|      |                                                                                                                                                             |     |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 4.4  | Vertical, horizontal, and diagonal correlation coefficients of adjacent pixels of encrypted images. . . . .                                                 | 110 |
| 4.5  | Vertical, horizontal, and diagonal correlation coefficients of encrypted Baboon image for the Ascon algorithm compared to different encryption methods. . . | 110 |
| 4.6  | Vertical, horizontal, and diagonal correlation coefficients of plaintext and cipher images. . . . .                                                         | 112 |
| 4.7  | Chi-square test values for cipher images produced by Ascon algorithm. . . .                                                                                 | 112 |
| 4.8  | Chi-square test values of Ascon algorithm compared to different encryption algorithms. . . . .                                                              | 112 |
| 4.9  | Maximum deviation results of Ascon algorithm compared to related algorithms.                                                                                | 113 |
| 4.10 | Irregular deviation results of Ascon compared to other image encryption algorithms. . . . .                                                                 | 113 |
| 4.11 | Deviation from uniform histogram of the Ascon algorithm compared to other encryption algorithms. . . . .                                                    | 114 |
| 4.12 | Global and local entropy values of images encrypted by Ascon algorithm. . .                                                                                 | 115 |
| 4.13 | Global entropy of cipher image Baboon of Ascon compared to different encryption algorithms. . . . .                                                         | 115 |
| 4.14 | Contrast analysis for plaintext and cipher images. . . . .                                                                                                  | 115 |
| 4.15 | Energy analysis for plaintext and cipher images. . . . .                                                                                                    | 116 |
| 4.16 | Homogeneity analysis for plaintext and cipher images. . . . .                                                                                               | 116 |
| 4.17 | Comparison of homogeneity, contrast, and energy values of cipher images of Ascon and related methods. . . . .                                               | 117 |
| 4.18 | MSE values of images encrypted by Ascon. . . . .                                                                                                            | 117 |
| 4.19 | MSE values for images encrypted by Ascon compared to related methods. . .                                                                                   | 118 |
| 4.20 | MAE values of images encrypted by Ascon. . . . .                                                                                                            | 118 |
| 4.21 | MAE values for images encrypted by Ascon compared to related encryption algorithms. . . . .                                                                 | 118 |

|                                                                                                  |     |
|--------------------------------------------------------------------------------------------------|-----|
| 4.22 PSNR results of images encrypted by Ascon. . . . .                                          | 119 |
| 4.23 PSNR of the images encrypted by Ascon compared to other encryption algo-<br>rithms. . . . . | 119 |
| 4.24 Avalanche effect $MSE_{av}$ values of images encrypted by Ascon. . . . .                    | 120 |
| 4.25 NPCR values of images encrypted by Ascon. . . . .                                           | 120 |
| 4.26 NPCR of Ascon compared with other encryption methods. . . . .                               | 121 |
| 4.27 UACI values of images encrypted by Ascon. . . . .                                           | 121 |
| 4.28 UACI of Ascon compared with other encryption methods. . . . .                               | 121 |
| 4.29 NIST SP 800-22 tests results for the encrypted Male image. . . . .                          | 125 |



# Acknowledgments

I begin with heartfelt thanks to the Almighty Allah for His abundant blessings. His grace has endowed me with the knowledge, patience, and determination that made this journey not only possible but also a monumental achievement in my life.

I am deeply indebted to my Ph.D. Advisor, Dr. Arslan Munir, for guiding me throughout my research process. His invaluable advice and encouragement played a crucial role in shaping the course of my work.

I extend my sincere gratitude to all the members of my committee. Their insightful suggestions during my RPE and proposal presentations helped my research with interesting ideas, that I was able to incorporate in my research.

I would also like to thank my family, colleagues and friends, for their support throughout this Ph.D. journey.

# Dedication

This work is dedicated to my mother, the pillar of my education and learning journey. Her devotion and care have been the guiding light that illuminated my path.

To my father, whose encouragement, support, and passion for technology paved the way for the path I am on today.

I am grateful for your boundless love, encouragement, and support that have been my constant companions.

To my accomplished sisters, Dr. Amal and Dr. Sahar, whose achievements have been a beacon of inspiration, driving me to strive for my highest potential.

To my dearest little sister, Nessreen, who has been more than a sibling, but my closest friend and companion.

To my beloved nephew, Bader, and my little bundle of joy, Fahad, who have filled my life with laughter and boundless happiness.

# Chapter 1

## Image Encryption Algorithms: A Survey of Design and Evaluation Metrics

### 1.1 Introduction

With the increased use of social media and share of multimedia over communication networks, image encryption has become a critical research topic in information security, where researchers aim to protect the integrity and confidentiality of the shared image data. Since multimedia data tends to have a high amount of redundancy due to inherent correlation between neighboring pixels, a need for a specialized and robust image encryption algorithm becomes essential. Since secure, widely used, and recommended encryption methods, such as Advanced Encryption Standard (AES), Twofish, RSA, etc. are not suitable for real-time multimedia data encryption due to high computational requirements and slow encryption speed, researchers have proposed various encryption techniques designed specifically for image encryption.

In this chapter, several proposed state-of-the-art image encryption algorithm techniques are reviewed with an emphasis on their strengths and weaknesses. This dissertation clas-

sifies image encryption algorithms into seven different approaches based on the techniques used. These approaches are: traditional ciphers, chaotic systems, DNA encoding, neural networks, compressive sensing, frequency domain, and meaningful sensing. Additionally, we have discussed in detail security, quality, and efficiency evaluation metrics, such as correlation coefficient, histogram analysis, entropy, mean square error (MSE), and NIST SP 800-22 Test, etc. This chapter aims to provide a brief introduction to image encryption algorithms for researchers and practitioners starting in this field and aid them in understanding the current state of image encryption algorithms. Our main contributions in this article are as follows:

- Classification of image encryption into seven approaches based on the techniques used in the algorithms.
- A thorough review of a comprehensive set of security, quality, and efficiency evaluation metrics for image encryption.
- Calculation of the upper and lower bounds for each of the evaluation metric for image encryption.

The rest of the chapter is organized as follows. Section 1.2 discusses the general concept of image encryption algorithms, and provides a detailed review of different approaches to image encryption algorithms. Section 1.3 presents the metrics used by researchers to evaluate image encryption algorithms. Finally, Section 1.5 concludes this chapter.

## 1.2 Image Encryption Algorithms

Digital images are electronic files that are made of two-dimensional arrays of numbers with variable sizes and number of channels (i.e., one channel for gray-scale images, three channels for color images, and four channels for color images with a transparency channel) that store pixel values. Each pixel represents a specific color. A digital image  $I$  is defined by a size  $H \times W \times nc$ , where  $H$  is the height,  $W$  is the width, and  $nc$  is the number of

channels. Images usually contain private and personal information, such as personal and legal documents, medical images, military images, etc. When these types of images are shared over the Internet or an unsecured network, these images could be compromised by unauthorized access. Furthermore, most digital images tend to contain high redundancy due to the correlation between neighboring pixels. Image encryption algorithms aim to ensure the security of the image from unauthorized access by encrypting the plaintext image and producing a cipher image that could only be decrypted by the intended users. Image encryption algorithms use the confusion and diffusion<sup>3</sup> principle to produce cipher images and reduce the correlation between neighboring pixels. Confusion is attained by changing the values of each pixel in a digital image by a substitution map, where the values of each pixel in a plaintext image are changed by a substitution map such as AES S-Box. Diffusion is achieved by changing the plaintext image pixels' location within an image itself using mathematical permutation operations. Diffusion helps in reducing the correlation between adjacent pixels that was present in the plaintext image. In image encryption, confusion and diffusion are controlled using a key or a set of keys to obscure and permute the plaintext image to produce the cipher image. Researchers have proposed a multitude of image encryption algorithms over the past decades. This section presents a review of image encryption algorithms based on the techniques used. Figure 1.1 illustrates the seven different approaches based on the techniques used.

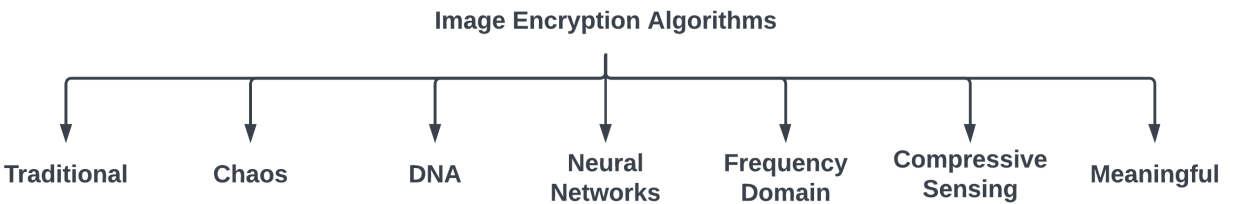


Figure 1.1: Different approaches to image encryption based on used techniques.

### 1.2.1 Traditional Ciphers

At the moment, traditional ciphers are the most commonly used algorithms to encrypt images over networks. The images are converted into bit streams, and then encrypted using

one of the traditional encryption methods.

The Advanced Encryption Standard (AES)<sup>4</sup> is a widely adopted and always recommended encryption algorithm. AES is a symmetric key encryption with variable key lengths and number of rounds. It uses 10 rounds of encryption for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys. AES works on blocks with a size of 128-bits and each round involves a series of SPN-based (substitution-permutation network-based) operations on each block of data, that is, SubBytes, ShiftRows, MixColumns, and AddRoundKey. The AES can be used in various modes, but the electronic codebook (ECB) mode is not suitable for image encryption as it produces cipher images with noticeable patterns where identical blocks of the image will result in producing identical blocks in the cipher image. To avoid this problem it is recommended to use AES in cipher block chaining (CBC) mode.

Twofish<sup>5</sup> is another symmetric key block cipher that has 16 rounds using variable key lengths of 128, 192, and 256 bits. Twofish uses Feistel network structure and works on 128-bits blocks. Each round involves a series of operations on each block of data, that is, substitution boxes, linear transform based on a Maximum Distance Separable (MDS) code, pseudo-Hadamard transform (PHT), and a key addition function.

Stream ciphers have also been used in image encryption. They produce pseudorandom keystream from a key and an initialization vector (IV) or nonce, which is then combined with the plaintext image in a bitwise operation (usually XOR) to produce the cipher image. Trivium<sup>1</sup> is a synchronous stream cipher that is based on a nonlinear-feedback shift register (NLFSR) which generates a key-stream of up to  $2^{64}$  bits using an 80-bit secret key and an 80-bit IV. Trivium cipher has three phases: initialization phase, warm-up phase, and encryption phase. In the initialization phase, the key and the IV are loaded into the 288-bit state register and all remaining bits are set to 0 except for three specific bits which are set to 1. In the warm-up phase, the cipher is clocked 1152 times without generating an output. In the encryption phase, all the bits produced after the 1152<sup>th</sup> cycle are used to generate the key-stream. To produce the cipher image, a key-stream equal to the size of the plaintext image is XORed with the plaintext image. ChaCha20<sup>6</sup> is another stream cipher that generates a key-stream of up to  $2^{64}$  bits using a 256-bit secret key, 64-bit nonce, and

a 64-bit counter. In ChaCha20 cipher, 512-bit state is divided into a 4 by 4 word matrix containing 8 key words, 4 constant words, and 4 words for the nonce and counter. ChaCha20 has 20 rounds where each round has 16 additions and 16 XORs and 16 constant-distance rotations of 32-bit words. After 20 rounds, a 512-bit key-stream is produced. To produce a cipher image, a key-stream equal to the size of the plaintext image is XORed with the plaintext image.

### 1.2.2 Chaotic Systems

Some of the most widely used image encryption algorithms are based on chaos theory due to several desirable properties, such as non-linearity, sensitivity to initial parameters, speed, and robustness<sup>7</sup>. In chaos-based algorithms, the pseudorandom chaotic sequences that are generated by the chaotic system are used to permute and diffuse a plaintext image<sup>8</sup>.

Some of the chaotic maps that are commonly utilized in image encryption methods are: Logistic map<sup>9;10</sup>, Baker map<sup>11;12</sup>, Arnold map<sup>13;14</sup>, Tent map<sup>15;16</sup>, and Hyperchaotic maps<sup>17;18</sup>, etc. Researchers have proposed a plethora of image encryption algorithms based on chaotic systems; the following is a brief overview of some of the proposed algorithms for each chaotic system.

#### Logistic Map

Logistic map is one of the simplest non-linear and deterministic dynamical systems that are used to generate chaotic sequences. The logistic map can be defined as:

$$X_{n+1} = rX_n(1 - X_n), \quad (1.1)$$

where  $X_0$  represents the initial parameter or starting value of the logistic map, with  $0 \leq X_i \leq 1$ . The parameter  $r$  is the control parameter or growth rate of the logistic map, and it has a range of  $(0, 4)$ . However, chaotic behavior is observed only when the control parameter  $r$  is within the range of  $(3.56995, 4)$ .<sup>19</sup> The bifurcation diagram of the logistic map is illustrated

in Figure 1.2.

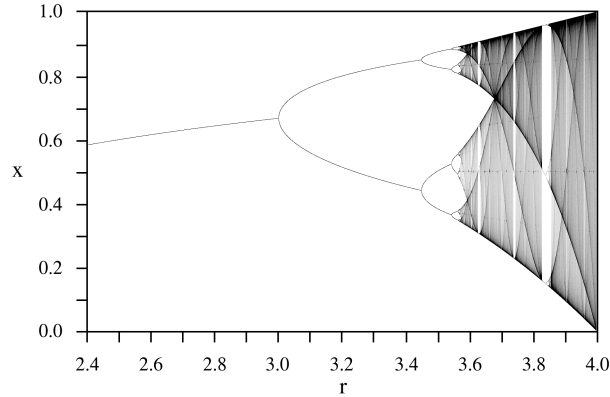


Figure 1.2: Logistic map bifurcation diagram.

Alghamdi et al.<sup>8</sup> have proposed an image encryption algorithm based on logistic map, permutations, and AES S-box. The proposed algorithm employs SHA-2 hash based on the plaintext image, a pre-shared key, and an initialization vector (IV) to generate the initial parameters for the logistic map. The pseudorandom sequences generated by the logistic map are used to chaotically permute and substitute the image at pixel-level.

In<sup>20</sup>, Rohith et al. have proposed an image encryption algorithm for medical applications using the key sequences of triple logistic maps. In the proposed algorithm, logistic map is utilized to generate three different key sequences  $X1_i$ ,  $X2_i$ , and  $X3_i$  with different initial values where  $X1_0 \neq X2_0 \neq X3_0$ . These key sequences are converted into discrete key sequences  $K1_i$ ,  $K2_i$ , and  $K3_i$  within the range of (0, 255). Then, a combined key sequence  $Ki$  is obtained through a bit-by-bit logical XOR operation between  $K1_i$ ,  $K2_i$ , and  $K3_i$ ; which is then used to encrypt the image by performing a logical XOR operation between image pixels and key sequence  $Ki$ . Logistic maps are popular in image encryption algorithms due to their minimal complexity and fast computation speed. However, a potential disadvantage is their limited range of chaotic behavior.

## Baker Map

The Baker map is a chaotic transformation that maps a unit square into itself. It involves a sequence of stretching and folding operations that result in a new transformed unit square.



The formula for the Baker map is:

$$B(x_{n+1}, y_{n+1}) = \begin{cases} (2x_n, y_n/2) & 0 \leq x < \frac{1}{2} \\ (2 - 2x_n, 1 - y_n/2) & \frac{1}{2} \leq x < 1 \end{cases}, \quad (1.2)$$

where  $(x_n, y_n)$  and  $(x_{n+1}, y_{n+1})$  are the coordinates of a point in the unit square before and after the transformation, respectively.

Elashry et al.<sup>11</sup> have proposed a design of a 2D chaotic Baker map that operates on three different modes: cipher block chaining (CBC) mode, cipher feedback (CFB) mode, and output feedback (OFB) mode. In the proposed algorithm, the Baker map performs the permutation procedure on the plaintext image by moving the positions of the pixels. The encryption key is a random IV which depending on the operation mode is XORed with the plaintext's data blocks, where the IV has the same size as the plaintext block. The proposed implementation has the speed of chaotic maps with the randomness of the mode of operation, resulting in a fast and secure cryptosystem.

In<sup>12</sup>, Mondal et al. have presented an image encryption algorithm that uses a 2D Baker map, where the plaintext image is initially permuted using a pseudorandom sequence generated by Baker map, followed by a diffusion process based on an XOR operation between the scrambled plaintext image and the generated sequence. The proposed encryption algorithm employs dual secret keys generated by the Baker map; one is used for permutation and the other for diffusion.

The Baker map is known to be vulnerable to chosen-plaintext attacks, where an attacker with a pair of plaintext and a corresponding cipher text can reverse the permutation and diffusion processes to deduce information about the secret key. This vulnerability could be mitigated by combining other layers of security, such as using an IV or using multiple chaotic maps.

## Arnold Maps

The Arnold map, which was introduced by Vladimir Arnold in 1968<sup>21</sup>, is a nonlinear chaotic transformation that maps a unit square into itself that exhibits chaotic behavior, which is desirable in image encryption for producing pseudorandom sequences. The Arnold map can be represented by the formula:

$$\Gamma(x_{n+1}, y_{n+1}) = (2x_n + y_n, x_n + y_n) \mod 1, \quad (1.3)$$

where  $(x_n, y_n)$  and  $(x_{n+1}, y_{n+1})$  are the coordinates of a point in the unit square before and after the transformation, respectively.

Rachmawanto et al.<sup>13</sup> have proposed an encryption algorithm based on Arnold map. The proposed algorithm divides the image into small blocks with the same width and height, and each of the small blocks is permuted using Arnold map. It is worth noting that the proposed algorithm does not diffuse the pixel's values of the Image, resulting in the cipher image having the same histogram information as the original plaintext image.

in<sup>14</sup>, Shalaby et al. have proposed a medical images encryption algorithm based on an enhanced Arnold map and AES. Their proposed algorithm consists of three steps. It starts by sharpening the edges of the plaintext image to preserve the details. Then the sharpened plaintext image's pixels are chaotically shuffled for  $i$  number of iterations based on a modified Arnold map. Finally, the permuted image is encrypted using AES -128 encryption in CBC mode.

It should be noted that Arnold map is designed to work only on square images where the height and width are equal, and is not suitable for use on rectangle images without first resizing the image to make it square before encryption. Furthermore, Arnold map is known to exhibit periodic behavior, meaning that after a certain number of permutation iterations, the image's pixels will return to their original position.

## Tent Maps

The Tent map is a chaotic system that maps real numbers in the range (0,1) to another real number in the same range. The Tent map can be represented by the formula:

$$x_{n+1} = f_{\mu}(x_n) = \begin{cases} \mu x_n & \text{if } x_n < \frac{1}{2} \\ \mu(1 - x_n) & \text{if } \frac{1}{2} \leq x_n \end{cases}, \quad (1.4)$$

where  $x_0$  is the starting value and  $x_{n+1}$  is the mapped value.  $\mu$  is the control parameter which has a range of (0, 2). However, chaotic behavior is observed only when the control parameter  $\mu$  is within the range of (1, 2). The bifurcation diagram for the Tent map is depicted in Figure 1.3. Chunhu et al. have presented an algorithm in<sup>15</sup> that utilizes a key-

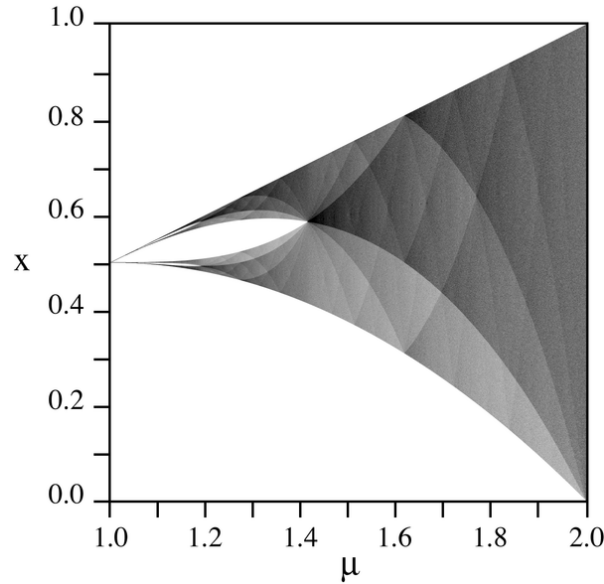


Figure 1.3: Tent map bifurcation diagram.

stream generated by Tent map. The key-stream is subsequently XORed with the plaintext image to produce the cipher image. Notably, the proposed algorithm does not perform any shuffling or permutation on the image's pixels.

In<sup>16</sup>, Vishwas et al. have proposed an image encryption algorithm based on Tent map. The proposed algorithm has two cases. In the first case, a chaotic sequence is generated based on the Tent map, and this sequence is then XORed with the plaintext image to produce the

cipher image. The second case is similar to the first case with the addition of a permutation step. The resulting cipher image is then permuted using the sequence generated from the Tent map in one of four variants (only rows are permuted, only columns, first rows then columns, or first columns then rows).

Tent map are easy to implement, have simple control of initial parameters, and require minimal computation power. However, they suffer from a small range of chaotic behavior similar to logistic map.

## Henon Maps

The Henon map is a two-dimensional map that maps a point  $(x_n, y_n)$  to a new point  $(x_{n+1}, y_{n+1})$ . Henon map can be defined as:

$$\begin{aligned} x_{n+1} &= 1 - ax_n^2 \\ y_{n+1} &= bx_n, \end{aligned} \tag{1.5}$$

where  $(x_n, y_n)$  and  $(x_{n+1}, y_{n+1})$  are the coordinates of a point before and after the mapping, respectively;  $a$  and  $b$  are the control parameters for the Henon map, however, the Map starts to behave chaotically when  $a = 1.4$ , and  $b = 0.3$ <sup>22</sup>. The bifurcation diagram for the Henon map is depicted in Figure 1.4. Pradeep et al.<sup>23</sup> proposed an image encryption algorithm

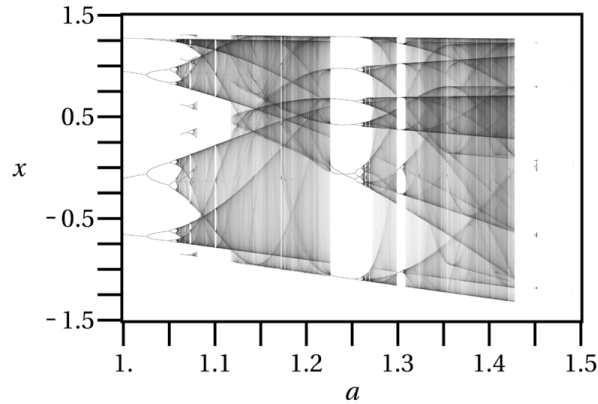


Figure 1.4: Henon map bifurcation diagram.

using Henon map. The proposed algorithm consists of three stages. In the first stage, Henon

map generates two sequences of random values. In the second stage, the first randomly generated sequence is XORed with the plaintext image to diffuse the pixels' values. Finally, the diffused image is permuted using the second sequence of values generated by the Henon map to produce the cipher image.

In<sup>24</sup>, Hussein et al. have introduced a new permutation-substitution algorithm based on Henon map. The algorithm utilizes Henon map to generate two sets of key-streams, namely X and Y, each having a length of  $(\text{height} \times \text{width} + 100)$ , where the first 100 generated values are removed from the key-stream. The plaintext image is then reshaped into a one-dimensional array, and the elements are permuted using the random numbers from key-stream X for two rounds. The Y key-stream is converted into a three-dimensional array to make it compatible with the three color channels (RGB) of the permuted array. Then, each color channel of the permuted array elements is diffused using an XOR operation with the corresponding Y three-dimensional key-stream.

## Hyperchaotic Systems

Hyperchaotic maps are high-dimensional nonlinear dynamical systems that are defined as any chaos system with at least two positive Lyapunov exponents<sup>25</sup>. Lyapunov exponent of a dynamical system is a parameter that measures the rate of separation of two infinitesimally close trajectories<sup>26</sup>. Hyperchaotic maps are considered to be better than simple chaotic maps due to their higher degree of non-periodicity and unpredictability<sup>27</sup>.

Gao have proposed an image encryption algorithm based on 2D hyperchaotic map in<sup>17</sup>. First, two hyperchaotic sequences x and y are generated by the 2D hyperchaotic map. Next, the plaintext image rows and column are permuted using chaotic sequence x. Finally, the pixel values of the permuted image are obscured through forward and backward diffusion using chaotic sequence y to produce the cipher image.

In<sup>18</sup>, Li et al. have proposed a hyperchaos based image encryption algorithm using pixel-level permutation, bit-level permutation and diffusion. First, two chaotic sequences are generated by a 5D multi-wing hyperchaotic system, where the control parameters are

related to the original image. Then, pixel-level permutation and bit-level permutation are performed on the plaintext image using the first chaotic sequence. Finally, the permuted image is diffused using the second chaotic sequence to produce the cipher image.

## Multiple Chaotic Maps

The security and complexity of a chaos-based image encryption based algorithm can be improved by using multiple chaotic maps instead of just one. Additionally, the limitation and vulnerabilities of one chaotic map can be mitigated by combining it with another chaotic map.

The study by Ramasamy et al.<sup>28</sup> have used an enhanced Logistic-Tent map for key generation in an image encryption algorithm based on block scrambling and modified zigzag transformation. The algorithm starts by dividing the plaintext image into four quadrants, and each quadrant is further divided into four sub quadrants. Next, each sub quadrant is rotated anti-clockwise by  $90^\circ$ . Then, each color channel of the scrambled image is further permuted using a modified zigzag transformation, where the upper left pixel and its next neighboring horizontal pixels are exchanged with the base right pixel. Finally, an XOR operation is performed between the permuted image and the key-stream generated by the enhanced Logistic Tent map.

Jain and Aji<sup>29</sup> have proposed an image encryption algorithm based on a 2-dimensional Logistic-Sine coupling map (2D-LSCM) and Arnold's cat map, where Arnold's cat map is used to shift the pixels of the plaintext image. Next, a chaotic matrix (matrix X) that has the same size of the plaintext image is generated by the 2D-LSCM chaotic map, and is used to perform another permutation on the pixels positions. The pixels values are then diffused using chaotic matrix X. This process of permutation and diffusion is repeated for two rounds to produce the cipher image.

### 1.2.3 DNA Encoding

DNA encoding can be used in image encryption by converting the image's pixel values into a DNA sequence. Since the DNA sequence is composed of four nucleotides, Adenine (A), Guanine (G), Cytosine (C), and Thymine (T), researchers can use DNA coding rules to convert the binary information into a DNA sequences. For example, a pixel with an RGB value (01010001,00101000,10001011) can be encoded as the DNA sequence GGAG,ACCA,CACT using rule 1 from the DNA coding rules in Table 1.1. Binary operation (i.e., addition, subtraction, XOR, etc.) can also work on DNA sequences using DNA operation rules tables. For example, the addition of the nucleotides  $A + A$  is A using rule 1. Since A in rule 1 is 00, the addition of  $00 + 00$  is 00, which is A. In the case of adding  $G + G$  using rule 1, the result is C, i.e  $01 + 01$  is 10 which is equivalent to C. Table 1.2 illustrates the DNA addition operation for the DNA rule 1 in Table 1.1.

Table 1.1: Sample of different DNA coding rules.

| Rule | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  |
|------|----|----|----|----|----|----|----|----|
| A    | 00 | 00 | 01 | 01 | 10 | 10 | 11 | 11 |
| T    | 11 | 11 | 10 | 10 | 01 | 01 | 00 | 00 |
| G    | 01 | 10 | 00 | 11 | 00 | 11 | 01 | 10 |
| C    | 10 | 01 | 11 | 00 | 11 | 00 | 10 | 01 |

Table 1.2: DNA addition for DNA rule 1.

| + | A | G | C | T |
|---|---|---|---|---|
| A | A | G | C | T |
| G | G | C | T | A |
| C | C | T | A | G |
| T | T | A | G | C |

The image encryption algorithm presented by Wu et al.<sup>30</sup> utilizes DNA coding in combination with a hyperchaotic map. In the proposed algorithm, a key-stream generated using Chen hyperchaotic map is used to scramble to the plaintext image. Afterwards, the resulting scrambled image and the key-stream are both DNA encoded. The algorithm then diffuses the DNA-encoded scrambled image with the DNA encoded key stream. Lastly, DNA decoding is performed on the resulting diffused image to produce the cipher image.

Li and Su<sup>31</sup> have proposed an algorithm based on Logistic map, Chen hyperchaotic map, and DNA encoding. The algorithm starts by dividing the image into three matrices based on the RGB color channels. Next, each of the three matrices are divided into blocks. The Chen hyperchaotic map is used to randomly select the DNA coding, decoding and operation rules of each block. Each block of the three color matrices is first DNA encoded, then goes through an add, subtraction, XOR or XNOR operation selected based on the Chen hyperchaotic map. The algorithm then performs DNA decoding on the blocks of the three color matrices followed by a row and column permutation based on logistic map. Finally, the three color matrices are combined into one image to obtain the cipher image.

In<sup>32</sup>, Li and Li have introduced a novel image encryption algorithm based on DNA dynamic encoding, Logistic-Tent map, and Lorenz hyperchaotic map. The proposed algorithm begins by dividing the plaintext image into rows, which are then DNA-encoded using rules selected by the Logistic-Tent map random sequence. The resulting DNA-encoded image is then scrambled using the sequence generated by Lorenz hyperchaotic map at DNA level. Next, the resulting scrambled DNA-encoded image is diffused using DNA operation rules selected based on the Logistic-Tent map. The resulting diffused image is then DNA decoded using the Lorenz hyperchaotic map. Finally, the resulting DNA-decoded image is diffused again using DNA operation rules selected based on the Logistic-Tent map to produce the cipher image.

The algorithm developed by Feng et al.<sup>33</sup> is based on DNA encoding, 2D-LSCM, SHA-256 hash of the plaintext image, and discrete logarithm. The proposed algorithm consists of three encryption steps: permutation and update, DNA sequence operation, and diffusion. All of the three steps are plaintext image-related. The algorithm begins with generating a SHA-256 hash of the plaintext image and then stretches the plaintext image into a 1D sequence. The algorithm also generates three integer chaotic sequences (ICSs) from the 2D-LSCM based on a secret key to be used in each step. For the permutation and update step, the algorithm permutes and updates the pixel values using the first generated ICS and the hash value using an equation based on modular and discrete logarithmic operations to calculate the permutation coordinates. Next, in the DNA sequence operation step, the



algorithm performs DNA encoding, DNA XOR, and then DNA decoding on the resulting permuted image utilizing DNA rules selected based on the second generated ICS and the hash value. For the diffusion step, the algorithm changes the pixel values in forward or backward diffusion based on the secret key, hash value, the three ICSs, and discrete logarithm operation. The three steps of the algorithm are performed twice, and the final 1D cipher sequence is converted into a 2D image to produce the cipher image.

### 1.2.4 Neural Networks

Neural networks can be utilized in image encryption algorithms to enhance the security and efficiency of the algorithm. Neural networks can be used to permute<sup>34</sup> and/or compress the plaintext images<sup>35</sup>. Furthermore, neural networks with chaotic behaviors<sup>36</sup> can be used to generate chaotic sequences which can be used as keys, input parameters, or to diffuse the image.

Man et al.<sup>34</sup> have proposed a double image encryption algorithm based on convolution neural network (CNN) and chaos. The proposed algorithm uses a Logistic map to generate the initial parameters for a 5D conservative chaotic system and CNN. Two plaintext images are first permuted by a plaintext-related chaotic pointer generated by the CNN. Next, each permuted image is split on the bit level into high and low 4-bit images. Then two high 4-bit images are fused into one image, and low 4-bit images are fused into one image. Finally, a dual channel encryption (optical encryption, and digital encryption) is performed on the two permuted and fused images to produce two cipher images.

An image encryption algorithm based on Hopfield chaotic neural network has been proposed by Wang and Li<sup>37</sup>. The proposed algorithm starts by generating the initial parameters for Arnold map and Hopfield neural network using a staged composite chaotic map based on a Logistic map and Tent map. Next, the plaintext image pixels are permuted by the random sequence generated by Arnold map. Then, Hopfield neural network is utilized to generate a self-diffusion chaotic key-stream matrix. Each color channel of the permuted image is then XORed with key-stream matrix to produce the cipher image.

Han et al.<sup>36</sup> have proposed an image encryption algorithm based on Hermite chaotic neural network. First, a chaotic training sample for the Hermite chaotic neural network is generated by Logistic map. Next, two sets of chaotic sequences are generated by the trained Hermite chaotic neural network. Finally, the two sets are summed and multiplied by 256 and then multiplied by the plaintext image to produce the cipher image.

In<sup>35</sup>, Yang et al. have proposed an image encryption algorithm based on back propagation neural network and a hyperchaotic system. First, the plaintext image is split into three matrices (R,G,B) and each matrix is decomposed into sub-image blocks. Next, the sub image blocks are provided as input to the back propagation neural network which produces three compressed pixel matrices. The compressed pixel matrices are then permuted using a zigzag confusion algorithm. After that, five chaotic sequences are generated by a fractional-order memristive hyperchaotic system and two of the sequences are selected randomly to be XORed with the permuted matrices. Finally, the three diffused matrices are combined to produce the cipher image.

Even though neural network-based image encryption algorithms offer secure encryption, they do have some disadvantages. The time and computational complexity are high for neural networks-based image encryption as neural networks require significant processing power and training time.

### 1.2.5 Frequency Domain

Using an image frequency domain in an image encryption algorithm can improve an algorithm's security. In frequency domain image encryption, instead of performing the encryption on the plaintext image directly in the spatial domain, the image is first transformed into the frequency domain, using Fourier transform (FT), wavelet transform (WT), Discrete Cosine Transform (DCT), etc., and the confusion and diffusion are performed in the frequency domain. The cipher image is then produced by inverse transforming the encrypted frequency domain into the spatial domain.

Ding et al.<sup>38</sup> have proposed an image encryption algorithm based on a fractional-order

Henon map, a 2D discrete wavelet transform (DWT), and a 4D hyperchaotic system. In the proposed algorithm, the plaintext image is transformed using DWT and then permuted using a chaotic sequence generated by the fractional-order Henon map. The 4D hyperchaotic system generates a chaotic sequence that is XORed with the permuted image to produce the cipher image.

Guan et al<sup>39</sup> have presented a chaotic image encryption algorithm using frequency domain DNA encoding. The proposed algorithm starts by transforming the plaintext image using fast Fourier transform (FFT) and obtain the amplitude phase, which are then reconstructed as 1D sequences. Next, four chaotic sequences are generated by a 4D hyperchaotic map and the DNA encoding, decoding and operation rules are generated using these chaotic sequences. After that, the amplitude and the phase sequences are DNA encoded, a DNA operations are performed , and then DNA decoded to diffuse and confuse the amplitude and phase. Finally, the cipher image is produced by performing an inverse fast Fourier transform (IFFT).

In<sup>40</sup>, Chen et al. have introduced a hybrid domain image encryption algorithm for gray-scale images based on 2-dimensional improved Henon map (2D-ICHM ), integer wavelet transform (IWT), bit plane decomposition, and DNA operations. The algorithm starts by generating a SHA-512 hash from the plaintext image which is used to derive the control parameters for the improved Henon map (2D-ICHM), which generates two chaotic sequences, namely X, Y. Then, the plaintext image is decomposed into eight binary bit planes , and the high bit planes are XORed with chaotic sequence X. Next, the IWT is applied to the XORed bit plane, and then scrambled using chaotic sequence X. After that, inverse IWT is performed followed by DNA encoding, DNA XOR operation, and DNA decoding. Finally, bidirectional diffusion is performed using chaotic sequences X and Y to produce the cipher image.

### 1.2.6 Compressive Sensing

In recent years, there has been a growing interest in compressive sensing-based image encryption algorithms. Compressive sensing-based algorithms have the advantage of simultaneously

sampling, compressing, and encrypting images, resulting in faster encryption time and a reduced image size. In a compressive sensing-based image encryption algorithm, a plaintext image is first transformed into a sparse domain using discrete transform. Then, the transformed image is compressed using a random measurement matrix. The measurement matrix is a matrix used to capture a reduced set of the essential measurements/information of the image's structure, which allows accurate compression and reconstruction from a reduced set of measurements. Finally, the compressed image is encrypted using permutation and diffusion techniques.

A novel image encryption algorithm based on compressive sensing, M sequence, and an improved 1D chaotic map based on Logistic, Sine and Chevyshev maps has been introduced by Dou and Li<sup>41</sup>. The proposed algorithm generates a SHA-512 hash from the plaintext image which is used to derive the control parameters for the improved 1D chaotic map and the M sequence linear-feedback shift registers (LFSR). The plaintext image is then transformed to a DWT and scrambled using the M sequence. Next, the improved 1D chaotic system is used to generate a measurement matrix that is used to compress and diffuse the scrambled image to produce the cipher image.

In<sup>42</sup>, Zhang et al. have proposed an image compression and encryption algorithm based on compressive sensing, Fourier transform, and chaotic maps. The proposed algorithm utilizes a Tent-Sine chaotic map to generate a measurement matrix using a SHA-256 hash of the plaintext image as initial parameters. Next, Arnold transform is used to permute the compressed image, then a 2D Fourier transform generated by Chen hyperchaotic map is used as a mask to produce the cipher image.

Wang et al.<sup>43</sup> have proposed an image encryption based on compressive sensing and DNA encoding. The algorithm utilizes four-wings hyperchaotic system and Kronecker product (KP). First, a chaotic sequence generated by the chaotic system is used to dynamically control the DNA coding. Afterwards, the measurement matrix is obtained by a four-wing chaotic system. Next, the plaintext image is sparsified and permuted by Fisher-Yates random scrambling. Finally, DNA coding and DNA XOR operation is performed to produce the cipher image.

### 1.2.7 Meaningful Encryption

Meaningful image encryption is a technique that utilizes image encryption and steganography to produce cipher images that have a visually meaningful image instead of a noise like image. There are two steps in a meaningful encryption algorithm. First the algorithm encrypts the image using an image encryption algorithm. The resulting encrypted image is then embedded in a reference image using a steganography algorithm.

Ping et al.<sup>44</sup> have proposed an image encryption algorithm utilizing compressive sensing, chaotic maps and steganography to produce meaningful cipher images. The proposed algorithm consists of two stages: a compressive sensing encryption step, and steganography step. For the compressive sensing encryption step, a 2D logistic-adjusted-Sine map (2D-LASM) and a 3D cat map are generated from a secret key. Then, the plaintext image is transformed into a sparse domain using DWT and permuted using the chaotic sequence generated by the 2D-LASM. The measurement matrix for compressive sensing is constructed using the chaotic sequences generated by the 3D cat map and the 2D-LASM. Finally, the measurement matrix is used to compress and encrypt the permuted image. In the steganography step, the resulting cipher image is embedded into a carrier image, and the order of the embedding position is encrypted using the chaotic sequence generated by the 2D-LASM.

In<sup>45</sup>, Wang et al. have proposed a meaningful image encryption based on chaotic map and random scrambling diffusion. The plaintext is first sparsified by wavelet packet transform (WPT). Next, a one-dimensional chaotic map is used to construct the measurement and permutation matrices. Then, a bidirectional random scrambling algorithm based on chaotic magic transformation (CMT) is used to permute the sparsified plaintext image. The permuted resulting image is compressed using the chaotic measurement matrix then diffused using chaotic pixel diffusion. Finally, the encrypted and compressed image is embedded in a carrier image using WPT embedding algorithm.

Jiang et al.<sup>46</sup> have proposed a novel meaningful image encryption algorithm based on parallel compressive sensing and slant transform. First, the plaintext image is sparsified by DWT and the sparse coefficient matrix of plaintext image is used as control parameters

for the 4D memristive hyperchaotic map which generates the scrambling matrix. Then, the sparsified plaintext image is permuted and compressed using Arnold scrambling and parallel compressive sensing to produce the cipher image. Finally, the cipher image is randomly embedded in a carrier image by slant transform-based embedding in a block-wise manner to produce a visually meaningful cipher image.

In <sup>47</sup>, Yang et al. have proposed a visually meaningful image encryption algorithm based on a new universal embedding model (UEM). The new UEM is used to embed the secret information into the host and be adaptive to different types of wavelet transform. First, the plaintext image is encrypted using an image encryption algorithms (the authors did not specify an exact algorithm) to produce a permuted and diffused image. Next, integer wavelet transform is performed on the carrier image to obtain the wavelet transform domain. Then, a four-dimensional discrete chaotic system is used to embed the permuted and diffused image in the integer wavelet sub-bands of the carrier image. Finally, inverse integer wavelet transform (iIWT) is performed on the resulting wavelet transform domain to produce a visually meaningful cipher image.

Meaningful image encryption algorithms have the advantage of keeping the embedded data secure even if the embedded image data was extracted via steganalysis tools. Since the original secret image is already encrypted and is a noise-like image, it adds another layer of security by preventing attackers from directly obtaining the secret image through steganalysis alone. However, these algorithms have a drawback of high computational requirements due to the implementation of two layers of security.

### 1.3 Evaluation Metrics

Researchers depend on a comprehensive set of metrics, such as correlation coefficient, histogram analysis tests (chi-square, maximum deviation, irregular deviation, and deviation from uniform histogram), information entropy (global and local), gray-level co-occurrence matrix (GLCM) analysis (contrast, energy, and homogeneity), encryption quality analysis (mean square error (MSE), mean absolute error (MAE), and peak signal-to-noise ratio

(PSNR)), resistance to differential attacks, number of pixels change rate (NPCR), unified average changing intensity (UACI), resistance to noise and data loss attacks, and the algorithm's key sensitivity, to evaluate and analyze image encryption algorithms.. Each of these tests are discussed in detail below. Figure 1.5 illustrates the most commonly used evaluation metrics for image encryption in literature.

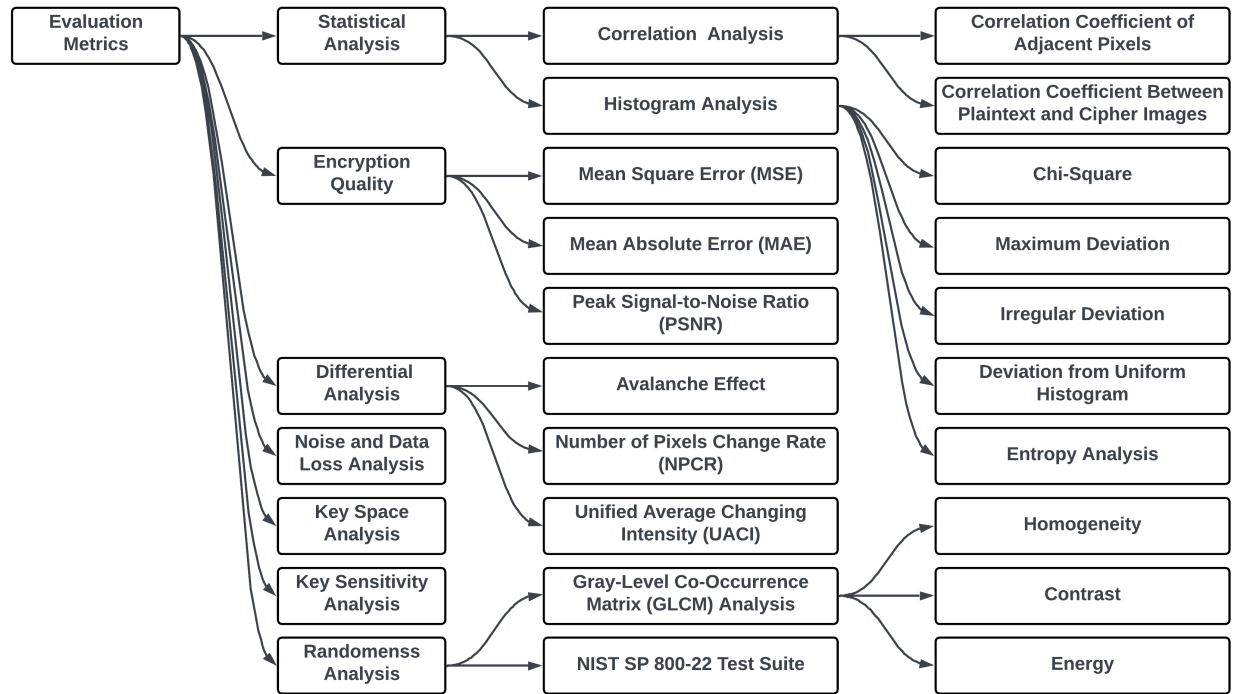


Figure 1.5: Most commonly used evaluation metrics.

### 1.3.1 Correlation Coefficient Analysis

Digital images typically exhibit strong correlations among adjacent pixels. To ensure a secure and robust image encryption algorithm, and to avoid vulnerability to statistical attacks, it is essential to eliminate this correlation. The correlation coefficient is a statistical test that is commonly used in image encryption to quantify the strength of correlation between two pixels. It has a range of  $(-1, 1)$ , where a value of 1 or  $-1$  indicates maximum positive or negative correlation respectively, while a value of 0 indicates no correlation between the compared pixels. Image encryption algorithms usually utilize the correlation coefficient metric to evaluate horizontal, vertical, and diagonal correlation coefficients between adjacent pixels

within a cipher image (Section 1.3.1) as well as between a plaintext image and its cipher (Section 1.3.1).

### Correlation Coefficient of Adjacent Pixels

The vertical correlation coefficient of adjacent pixels can be calculated as:

$$CC_v = \frac{\sum_{i=1}^{H-1} \sum_{j=1}^W (C_{(i,j)} - \bar{C})(C_{(i+1,j)} - \bar{C})}{\sqrt{\sum_{i=1}^{H-1} \sum_{j=1}^W (C_{(i,j)} - \bar{C})^2 \sum_{i=1}^{H-1} \sum_{j=1}^W (C_{(i+1,j)} - \bar{C})^2}}, \quad (1.6)$$

where  $H$  and  $W$  are an image's height and width, respectively.  $C_{(i,j)}$  and  $C_{(i+1,j)}$  refer to the pixel values of two adjacent pixels in the cipher image at positions  $(i, j)$  and  $(i + 1, j)$ , respectively. The mean pixel value of the cipher image is denoted by  $\bar{C}$ . Furthermore, to calculate the horizontal correlation coefficient, Equation (1.6) is adjusted such that the value of the pixel at  $C_{(i,j+1)}$  is used instead of  $C_{(i+1,j)}$ , and the value of pixel at  $C_{(i+1,j+1)}$  is used to calculate the diagonal correlation coefficient.

### Correlation Coefficient Between Plaintext and Cipher Images

The correlation coefficient between a plaintext image and its cipher image can be calculated as:

$$CC_{P,C} = \frac{\sum_{i=1}^H \sum_{j=1}^W (P_{(i,j)} - \bar{P})(C_{(i,j)} - \bar{C})}{\sqrt{\sum_{i=1}^H \sum_{j=1}^W (P_{(i,j)} - \bar{P})^2 \sum_{i=1}^H \sum_{j=1}^W (C_{(i,j)} - \bar{C})^2}}, \quad (1.7)$$

where  $P_{(i,j)}$  and  $C_{(i,j)}$  are the values of the pixel at index  $i, j$  of the plaintext image and the cipher image, respectively, and  $\bar{P}$  and  $\bar{C}$  represent the mean of the pixel values of the plaintext image and the cipher image, respectively.

## 1.3.2 Histogram Analysis

Histogram analysis is used to ensure the security and robustness of image encryption algorithms by evaluating the uniformity of a cipher image's histogram. The histogram of an



image is an important indicator that reflects the distribution of its pixel values. A cipher image with a close to uniformly distributed histogram prevents attackers' attempts to extract any useful information from the cipher image. Researchers in image encryption use a set of metrics to quantify histogram analysis, such as chi-square ( $\chi^2$ ), maximum deviation, irregular deviation, and deviation from uniform histogram. Each of the aforementioned metrics is further discussed in the following subsections.

### Chi-Square ( $\chi^2$ )

Chi-square  $\chi^2$  is a statistical test that is used in image encryption to test the randomness and uniformity of cipher images produced by an algorithm, by comparing the cipher image's histogram values with the expected distribution of a uniform histogram<sup>48</sup>. Chi-square tests each pixel of the cipher image to determine whether the distribution of pixels' values is uniform or not. Chi-Square ( $\chi^2$ ) is mathematically represented as:

$$\chi^2 = \sum_{i=0}^{255} \frac{(f_i - \mathcal{E})^2}{\mathcal{E}} \quad (1.8)$$

$$\mathcal{E} = \frac{H \times W}{256},$$

where  $f$  represents the histogram of the cipher image,  $f_i$  is the cipher image's histogram value at index  $i$ ,  $\mathcal{E}$  represents the expected distribution value of a uniform histogram, and  $H$  and  $W$  are the height and width of the cipher image, respectively. A uniformly distributed histogram has a  $\chi^2$  value of 0. When the  $\chi^2$  value is low, it indicates that the histogram distribution of the cipher image is close to a uniform distribution. Furthermore, when the  $\chi^2$  value is higher then the histogram is non-uniform. Since the histogram of an image is calculated for the pixel values ranging from 0 to 255, we have, thus, 255 degrees of freedom, and if we assume a significance level of 0.05, then the critical chi-square value is 293.24. If the cipher image  $\chi^2$  is lower or equal to the critical chi-square value, we accept the result as close to being uniform. From equation (1.8) we can calculate the lower and upper limit of  $\chi^2$ . To find the lower limit, we assume that the image has a uniform histogram, then each

element of the histogram  $f$  will have a value of  $\frac{H \times W}{256}$ . By substituting  $f$  in equation (1.8), the lower limit can be calculated as follows:

$$\begin{aligned}
\chi^2 &= 256 \times \frac{(\frac{H \times W}{256} - \mathcal{E})^2}{\mathcal{E}} \\
&= 256 \times \frac{(\frac{H \times W}{256} - \frac{H \times W}{256})^2}{\frac{H \times W}{256}} \\
&= 256 \times \frac{(0)^2}{\frac{H \times W}{256}} \\
\chi^2 &= 0,
\end{aligned} \tag{1.9}$$

From equation (1.9) the lower limit for  $\chi^2$  is 0. the upper limit, we assume that the image has one color, then the histogram  $f$  will have one element with value  $H \times W$  and the rest will be zeros. By substituting  $f$  in equation (1.8), the upper limit can be calculated as follows:

$$\begin{aligned}
\chi^2 &= (255 \times \frac{(0 - \mathcal{E})^2}{\mathcal{E}}) + \frac{(H \times W - \mathcal{E})^2}{\mathcal{E}} \\
&= 255 \times \mathcal{E} + \frac{(H \times W - \mathcal{E})^2}{\mathcal{E}},
\end{aligned} \tag{1.10}$$

Table 1.3 utilizes equation (1.10) to compute the upper limit of  $\chi^2$  for different sized images.

Table 1.3: Upper limit of Chi-square  $\chi^2$  for different sized images.

| Size        | 128×128   | 256×256    | 512×512    | 1024×1024   |
|-------------|-----------|------------|------------|-------------|
| Upper limit | 4,177,920 | 16,711,680 | 66,846,720 | 267,386,880 |

## Maximum Deviation

Maximum deviation ( $D_{max}$ ) is a metric used to evaluate the uniformity of the cipher image's histogram by measuring the deviation between the histograms of a plaintext image and its cipher. The maximum deviation metric helps to determine the quality of the image encryption algorithm by testing whether the produced cipher image differs significantly from the original image. A higher value of the maximum deviation indicates a better encryption

quality, since the algorithm produces a cipher image that is more deviated from the plaintext image. The mathematical expression of maximum deviation can be given as:

$$D_{max} = \frac{d_0 + d_{255}}{2} + \sum_{i=1}^{254} d_i \quad (1.11)$$

$$d = histogram(|P - C|),$$

where  $d$  is the histogram of the absolute values of the difference between the plaintext image and its cipher,  $d_i$  represents the histogram  $d$  value at index  $i$ , and  $d_0$  and  $d_{255}$  are the histogram values at index 0 and 255, respectively.

From equation (1.11) we can calculate the upper limit of  $D_{max}$ . If we assume that both images are completely different from each other,  $d$  will have one element with value  $H \times W$  and the rest will be zeros. By substituting  $d$  in equation (1.11), the upper limit can be calculated as follows:

$$D_{max} = \frac{(0 + H \times W)}{2} + (254 \times 0)$$

$$= \left(\frac{H \times W}{2}\right), \quad (1.12)$$

The lower limit of  $D_{max}$  is 0 for any size image. Table 1.4 utilizes equation (1.12) to compute the upper limit for different sized images.

Table 1.4: Upper limit of maximum deviation for different sized images.

| Size        | 128×128 | 256×256 | 512×512 | 1024×1024 |
|-------------|---------|---------|---------|-----------|
| Upper limit | 8,192   | 32,768  | 262,144 | 524,288   |

## Irregular Deviation

The irregular deviation ( $D_{irregular}$ ) is another metric used in image encryption to evaluate the quality of an algorithm by measuring the difference between the histograms of a plaintext image and its cipher<sup>49</sup>. Unlike the maximum deviation metric, irregular deviation also considers the difference between neighboring pixels. A cipher image with a lower value of  $D_{irregular}$  indicates better encryption algorithm quality. The irregular deviation can be

mathematically expressed as follows:

$$D_{irregular} = \sum_{i=0}^{255} [d_i - D_{avg}], \quad (1.13)$$

where

$$D_{avg} = \frac{1}{256} \sum_{i=0}^{255} d_i, \quad (1.14)$$

where  $d_i$  represents the histogram value at index  $i$  (from Equation (1.11)), and  $D_{avg}$  denotes the average value of the pixels that are deviated at every deviation value.

From equation (1.13), the upper limits of  $D_{irregular}$  can be calculated. If we assume that both images are completely different from each other,  $d$  will have one element with value  $H \times W$  and the rest will be zeros. By substitution  $d$  in equation (1.13), the upper limit can be calculated as follows:

$$\begin{aligned} D_{irregular} &= \left| H \times W - \left( \frac{H \times W}{256} \right) \right| + (255 \times \left| 0 - \frac{H \times W}{256} \right|) \\ &= \left| H \times W - \left( \frac{H \times W}{256} \right) \right| + \left( \frac{255 \times H \times W}{256} \right), \end{aligned} \quad (1.15)$$

The lower limit of  $D_{irregular}$  is 0 for any size image. Table 1.5 utilizes equation (1.15) to compute the upper limit for different sized images.

Table 1.5: Upper limit of irregular deviation for different sized images.

| Size        | 128×128 | 256×256 | 512×512 | 1024×1024 |
|-------------|---------|---------|---------|-----------|
| Upper limit | 32,640  | 130,560 | 522,240 | 2,088,960 |

## Deviation from Uniform Histogram

The deviation from uniform histogram ( $D_{uniform}$ ) is a metric used to measure the deviation of a cipher image's histogram from a uniform distribution. A cipher image with a histogram distribution close to a uniform distribution indicates good quality of the employed encryp-

tion algorithm<sup>49</sup>. When a cipher image produced by an image encryption algorithm has a histogram distribution that is close to uniform distribution, it implies that each pixel value occurs with equal frequency. The deviation from the uniform histogram metric can be mathematically expressed as follows:

$$D_{uniform} = \frac{\sum_{i=0}^{255} |H_{C_i} - H_{U_i}|}{H \times W}, \quad (1.16)$$

$$H_{U_i} = \begin{cases} \frac{H \times W}{256} & \text{if } 0 \leq i \leq 255 \\ 0 & \text{if } 0 > i > 255 \end{cases}$$

where  $H_{C_i}$  refers to the histogram value of the cipher image at index  $i$ , and  $H_{U_i}$  represents the histogram value of the uniform distribution at index  $i$ .  $H_{U_i}$  values are only considered when the pixel value is between (0–255), otherwise,  $H_{U_i}$  is set to zero. From equation (1.16), we can calculate the upper limit of  $D_{uniform}$ . If we assume that an image is extremely deviated from a uniform histogram,  $H_C$  will have one element with value  $H \times W$  and the rest will be zeros. By substitution  $H_C$  in equation (1.16), the upper limit can be calculated as follows:

$$\begin{aligned} D_{uniform} &= \frac{|H \times W - (\frac{H \times W}{256})| + (255 \times |0 - \frac{H \times W}{256}|)}{H \times W} \\ &= \left| \frac{H \times W}{H \times W} - \frac{\frac{1}{256} \times H \times W}{H \times W} \right| + \frac{\frac{255 \times H \times W}{256}}{H \times W} \\ &= \left| 1 - \frac{1}{256} \right| + \frac{255}{256} \end{aligned} \quad (1.17)$$

$$D_{uniform} = 1.9921875$$

The lower limit of  $D_{uniform}$  is 0 for any size image, and from equation (1.17) It is clear that the maximum possible  $D_{uniform}$  value for any image regardless of its size is 1.9921875.

### 1.3.3 Entropy Analysis

Information entropy is a statistical test that estimates the uncertainty and randomness in communication systems. It was first introduced by Claude Shannon in 1948<sup>50</sup>. In image encryption, entropy is used to measure the randomness of the distribution of pixels in the cipher image. A high entropy value indicates a better obfuscation of the plaintext, and the absence of any discernible pattern. To evaluate the unpredictability and randomness of an encryption algorithm, researcher rely on two types of entropy tests, namely, global entropy and local entropy. Shannon entropy, also known as global entropy, calculates the pixel information for the full image, while local entropy measures the mean entropy of randomly selected non-overlapping blocks. Shannon entropy can be calculated as:

$$H(m) = - \sum_{i=0}^{2^n-1} p(m_i) \log_2[p(m_i)], \quad (1.18)$$

where  $n$  represents the number of bits used to represent the symbol  $p(m_i)$ , and  $p(m_i)$  represents the probability of symbol  $m_i$ . Entropy for an image can be calculated using Equation (1.18), where  $p(m_i)$  represents normalized histogram counts for each intensity value in the image. Given that a pixel's maximum possible intensity value in an 8-bit representation is 256, and that the probability of a pixel value occurrence is  $\frac{1}{256}$ . The ideal value of entropy for an image can be calculated using equation (1.18) as follows:

$$H_{ideal} = - \sum_{i=0}^{255} \frac{1}{256} \times \log_2 \frac{1}{256} = 8$$

The entropy of a cipher image generated by an encryption algorithm should be as close as possible to the optimal entropy value of 8 to withstand brute force attacks.

The local entropy is measured by computing the mean Shannon entropy of randomly selected non-overlapping blocks. Local entropy is considered to be more accurate, consistent,

and efficient than global Shannon entropy<sup>51</sup>. Local entropy can be calculated as:

$$H_{K,T_B}(S) = \sum_{i=1}^K \frac{H(S_i)}{K}, \quad (1.19)$$

where  $S$  is a set of non-overlapping blocks selected randomly containing  $T_B$  pixels, and  $K$  represents the number of random blocks.  $H(S_i)$  is the Shannon entropy (from Equation (1.18)) of the  $i$ th block.

### 1.3.4 Gray-Level Co-Occurrence Matrix (GLCM) Analysis

The GLCM is a statistical test used in image encryption to analyze the texture of an image by considering the spatial relationship of pixels. It consists of a two-dimensional matrix of joint probabilities between pairs of pixels with specific values. By calculating the frequency of a pixel with gray-level value  $i$  occurring horizontally adjacent to a pixel with gray-level value  $j$ , the GLCM is useful for feature extraction and texture analysis. From the GLCM, statistical measures, namely homogeneity, contrast, and energy can be extracted to provide information about the image's texture features. In the following, we further discuss each of these measurements.

#### Homogeneity

In image encryption, homogeneity analysis is used to assess the uniformity of texture in a cipher image. It measures the closeness of the elements in the GLCM distribution to the GLCM diagonal<sup>52</sup>. Homogeneity values range between (0, 1), where a cipher image with high homogeneity indicates that similar pixels are close to each other, which may reveal information about the plaintext image. On the other hand, lower homogeneity values indicate better encryption quality. Where a cipher image produced by an image encryption algorithm with low homogeneity has a more randomized pixels distribution. The homogeneity formula is as follows:

$$Homogeneity = \sum_{i,j} \frac{p(i,j)}{1 + |i - j|}, \quad (1.20)$$

where  $i$  and  $j$  represent two gray-level values in the GLCM, And  $p(i, j)$  denotes the probability of  $i$  and  $j$  occurring horizontally adjacent to each other in the image, and is given by the value of the element at position  $(i, j)$  in the normalized GLCM.

## Contrast

Contrast analysis assesses the differences in pixel intensities between a pixel and its neighboring pixels in a cipher image<sup>52</sup>. A high contrast value between neighboring pixels indicates a strong encryption algorithm that is effective in hiding the plaintext image's features. Furthermore, a high degree of contrast between neighboring pixels indicates greater randomness in the cipher image. The contrast can be calculated as follows:

$$Contrast = \sum_{i,j} |i - j|^2 p(i, j), \quad (1.21)$$

## Energy

Energy analysis is used as a metric to describe the texture of a cipher image. It is calculated by summing the squared elements in the GLCM, and the resulting energy values range between  $(0, 1)$ <sup>52</sup>. Low energy values in the produced cipher images indicate that the algorithm has successfully randomized the pixels of the plaintext image. The energy can be calculated from the following expression:

$$Energy = \sum_{i,j} p(i, j)^2, \quad (1.22)$$

### 1.3.5 Encryption Quality

The quality of an image encryption algorithm is analyzed using various metrics, by comparing the pixel values of plaintext images and their respective ciphers. The encryption algorithm is considered to have a good quality when there is a high change in pixel values between plaintext images and their respective ciphers. To quantify the encryption quality, image encryption researchers use a set of metrics, namely, MSE, MAE, and PSNR. Each of these metrics is discussed further in the following.



## Mean Square Error

The MSE is a metric used to calculate the average square difference between the pixel values of a plaintext image and its cipher. In image encryption, it is used to evaluate the quality of an image encryption algorithm<sup>53</sup>. A lower MSE value indicates a greater similarity between the plaintext image and its cipher, whereas a higher MSE values means lower similarity between the plaintext image and its cipher, and therefore, a higher MSE value signifies a higher quality encryption. The MSE between a plaintext image and its cipher image can be calculated as follows:

$$MSE = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W [P(i, j) - C(i, j)]^2, \quad (1.23)$$

where  $H$  and  $W$  represent the height and width of the images, respectively.  $P(i, j)$  and  $C(i, j)$  denote the pixel values of the plaintext image and the cipher image at position  $(i, j)$ , respectively. From equation (1.23), the minimum and maximum possible MSE value for an image regardless of its size can be calculated. For the minimum value of MSE, if we assume the two images to be identical then  $P(i, j) - C(i, j)$  will be 0. Thus, the MSE minimum value is 0. For the maximum value, we assume that the two images are completely different from each other (one with 255 pixel values, and the other with 0 pixel values), the maximum possible MSE can be calculated as follows:

$$\begin{aligned}
MSE &= \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W [255 - 0]^2 \\
&= \frac{1}{H \times W} \times H \times W \times [255]^2 \\
&= \frac{H \times W \times [255]^2}{H \times W}
\end{aligned} \tag{1.24}$$

$$MSE = 65,025$$

### Mean Absolute Error

The MAE is another statistical metric used to evaluate the quality of an image encryption algorithm. It measures the average absolute difference between the pixel values of a plaintext image and its cipher image<sup>54</sup>. A lower MAE value indicates a greater similarity between the plaintext image and its cipher, whereas a higher MAE value indicates a lower similarity between the plaintext image and its cipher, and therefore, a higher MAE values means a higher quality encryption. The MAE can be calculated as follows:

$$MAE = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W |P(i, j) - C(i, j)|, \tag{1.25}$$

From equation (1.25), the minimum and maximum possible MAE value for an image regardless of its size can be calculated. For the minimum value of MAE, if we assume the two images to be identical then  $P(i, j) - C(i, j)$  will be 0. Thus, the MAE minimum value is 0. For the maximum value, we assume that the two images are completely different from each other (one with 255 pixel values, and the other with 0 pixel values), the maximum possible MAE can be calculated as follows:

$$\begin{aligned}
MAE &= \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W [255 - 0] \\
&= \frac{1}{H \times W} \times H \times W \times [255] \\
&= \frac{H \times W \times [255]}{H \times W}
\end{aligned} \tag{1.26}$$

$$MAE = 255$$

### Peak Signal-to-Noise Ratio

In image encryption, the PSNR metric is used to measure the noise ratio between the plaintext image and its cipher image<sup>7</sup>. A PSNR value of 0 indicates that the tested image is equivalent to random noise, and a higher value means a higher quality image which is closer to the plaintext image. The lower the PSNR value in the cipher images produced by an encryption algorithm, the better the quality of the encryption. PSNR can be calculated as follows:

$$PSNR = 10 \log_{10} \frac{MAX_p}{MSE}, \tag{1.27}$$

where  $MSE$  is the mean square error value which is calculated using equation (1.23), and  $MAX_p$  is the maximum value a pixel can have (which is typically 255 in 8-bit pixels).

### 1.3.6 Resistance Against Differential Attacks

Cryptanalysts use differential attacks to find the relationship between a plaintext image and its cipher by making small alterations to the plaintext image and encrypting it with the same key. An encryption algorithm should have excellent diffusion characteristics, where modifying a single pixel in the plaintext image should generate a completely different cipher image. An encryption algorithm's ability to resist differential attacks can be evaluated by

analyzing the algorithm's performance using avalanche effect, NPCR, and UACI tests. The details of each of these tests are further discussed in the following.

### **Avalanche Effect**

In image encryption, the avalanche effect refers to the phenomenon that occurs where a slight change to the key or the plaintext image results in an image encryption algorithm producing a significantly different cipher image. An encryption algorithm is considered to be safe against differential attacks when one pixel change in the plaintext image or key results in a change of more than 50% of the pixels in the cipher image<sup>55</sup>. The avalanche effect can be analyzed using the MSE metric by modifying equation (1.23) to calculate the mean squared difference between the two cipher images which are produced by encrypting a plaintext image, and the same plaintext image with only one pixel modified. The avalanche effect MSE ( $MSE_{av}$ ) can be calculated as follows:

$$MSE_{av} = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W [C_1(i, j) - C_2(i, j)]^2, \quad (1.28)$$

where  $C_1$  and  $C_2$  are the two cipher images produced by encrypting two plaintext images, where the two plaintext images differ in only one pixel, with the remaining pixels being the same.  $C_1(i, j)$  and  $C_2(i, j)$  represent the pixel values at position  $(i, j)$  of  $C_1$  and  $C_2$ , respectively. The minimum value of  $MSE_{av}$  is 0, and the maximum value is 65,025, same as the MSE minimum and maximum values, which were calculated in Section (1.3.5).

### **Number of Pixels Change Rate (NPCR)**

The NPCR metric measures the percentage of the differences between two cipher images that are produced by encrypting two plaintext images that differ in only one pixel with all other pixels being the same in the two plaintext images<sup>56</sup>. A higher NPCR value indicates a greater responsiveness from the algorithm to changes in the plaintext image or key, and thus better resistance to differential attacks. In other words, a good encryption algorithm

should be extremely sensitive to minor changes in the key or the plaintext image to resist such attacks. Wu et al.<sup>57</sup> have calculated that the ideal NPCR value as  $\geq 99.6094\%$ . The NPCR value can be calculated as:

$$NPCR = \sum_{i=1}^H \sum_{j=1}^W \frac{D(i,j)}{H \times W} \times 100\%, \quad (1.29)$$

where

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

where  $D(i,j)$  is the sum of the number of different pixels between  $C_1$  and  $C_2$  at position  $(i,j)$ .  $D(i,j)$  has a value of 1 when the pixel values of  $C_1$  and  $C_2$  at position  $(i,j)$  are different, otherwise, it has a value of zero.

### Unified Average Changing Intensity (UACI)

The UACI is another metric that is used to assess an encryption algorithm's ability to resist differential attacks by measuring the average change intensity between two cipher images that are produced by encrypting two plaintext images that are the same but differ in only one pixel<sup>56</sup>. Same as the NPCR, a higher UACI value indicates a better resistance to differential attacks. The ideal value for UACI is  $\geq 33.4635\%$  as calculated by Wu et al.<sup>57</sup>. The UACI value can be calculated as:

$$UACI = \frac{1}{H \times W} \left[ \frac{\sum_{i=1}^H \sum_{j=1}^W |C_1(i,j) - C_2(i,j)|}{255} \right] \times 100\%, \quad (1.30)$$

### 1.3.7 Resistance to Noise and Data Loss

Digital images transmitted over communication networks are often affected by noise and data loss, which can render cipher images undecryptable. Such distortions could also be introduced by attackers aiming to corrupt cipher images. These attacks are known as noise and occlusion attacks. To ensure the reliability of an image encryption algorithm, its ability

to withstand such attacks and decrypt the plaintext image from the altered cipher image needs to be tested<sup>58</sup>. To evaluate an encryption algorithm’s resilience against noise attacks, researchers introduce salt and pepper, and Gaussian noise to the cipher image with variable densities and try to decrypt the affected cipher images. For resistance against occlusion attacks, researchers evaluate the encryption algorithm’s ability to decrypt a cipher image that had segments cropped out at varying degrees of severity. Furthermore, the PSNR metric (section 1.3.5) could be utilized to quantify the quality of the encryption algorithms at decrypting a corrupted image by comparing the PSNR of the decrypted corrupted image to the plaintext image<sup>59</sup>. A higher PSNR of the decrypted image when compared to the original plaintext image indicates that the decrypted corrupted image is closer to the original plaintext image.

### 1.3.8 Key Space and Key Sensitivity

The key space represents the set of all possible keys that an encryption algorithm can use. To ensure a stronger level of security against brute force attacks, the key space of an image encryption algorithm should be larger than  $2^{100}$ <sup>60</sup>.

Key sensitivity analysis assesses the image encryption algorithm’s sensitivity to small changes to the key. A robust and secure algorithm should produce two completely different cipher images when encrypted by two keys that are different by only one bit. Furthermore, it should not be able to decrypt a cipher image and retrieve the plaintext image with a key that has one bit altered<sup>61</sup>.

To evaluate an encryption algorithm’s key sensitivity for encryption, researchers encrypt a plaintext image with two keys  $k_1$  and  $k_2$  that differ in only one bit and compare the resulting cipher images. For the case of evaluating key sensitivity for decryption, researchers attempt to decrypt a cipher image that was encrypted with key  $k_1$  using key  $k_2$ , which differs in only one bit. If the algorithm is sensitive to changes in the key, it should not be able to decrypt the cipher image with the altered key.

### 1.3.9 NIST SP 800-22 Test Suite

The NIST SP 800-22 test suite is a set of statistical tests developed by the National Institute of Standards and Technology (NIST)<sup>62</sup>. It is used by researchers to evaluate the randomness characteristics and strength properties of a sequence to determine if they are sufficiently random. The test suite consists of 15 statistical tests, each of which generates a p-value in the range  $(0, 1)$ , where a p-value of 0 indicates that the sequence is not random at all, and a p-value of 1 indicates that the sequence is completely random. In image encryption, the test suite is used to evaluate the randomness of the produced cipher image. For conducting these NIST tests, the image must be first transformed into a bit stream, after which the test suite can process the data and produce results for each test. If the p-value of a test is greater than a significance level  $\alpha = 0.01$ , then the null hypothesis is accepted, that is, the sequence is considered to be random and passes the test. If the computed p-value is less than 0.01, then the null hypothesis is rejected, that is, the sequence appears to be non-random.

## 1.4 Discussion

In this section, the advantages, disadvantages and the applications of each of the aforementioned image encryption techniques are explored. Each image encryption technique provides a level of security and privacy, yet each of the technique has some pros and cons.

Traditional encryption algorithms are still the most commonly recommended and used algorithms to encrypt images over networks. Their security has been assessed and proven through standardized testing and cryptanalysis. Their downside is that they typically require extensive computation time and significant processing power. Furthermore, traditional encryption algorithms fail when attempting to decrypt cipher images that have some corrupted pixels, since the decrypted data must be the exact plaintext data. Traditional encryption produces faulty decrypted images when decrypting a cipher image that has been affected by noise or has lost pixels, whether due to attacks or transmission errors. In multimedia applications, the content of an image is important and minor distortions in the original image

are acceptable<sup>63</sup>.

Chaotic systems-based image encryption algorithms provide excellent security due to their high sensitivity to the initial parameters, fast computation speed, and minimal complexity. A potential disadvantage of chaotic systems-based image encryption algorithms is the limited range of chaotic behavior for some chaotic maps such as the Logistic and Tent maps. Moreover, certain chaotic maps, such as the Baker map, have been known to be vulnerable to chosen-plaintext attacks. This vulnerability could be mitigated by adding other layers of security, such as utilizing an IV or nonce, or incorporating multiple chaotic maps. Furthermore, some chaotic maps are known to exhibit periodic behavior, meaning that after a certain number of permutation iterations, the image's pixels will return to their original position.

DNA-based image encryption is an innovative concept with high security levels; however it is still largely experimental and primarily explored in academic research. It has not yet found widespread practical application in data encryption scenarios. Furthermore, it requires specialized laboratory equipment to handle DNA synthesis and sequencing. This level of complexity hinders the practical application of DNA-based image encryption.

Neural networks-based image encryption algorithms have primarily emerged due to advancements in artificial neural networks and deep learning. Once effectively trained, they can provide a quick means of image encryption. However, the level of security they offer remains a relatively unexplored domain. Further research and analysis on the security of neural networks-based image encryption algorithms are imperative to comprehensively understand and quantify the robustness of these algorithms against potential threats and attacks.

Frequency domain-based image encryption techniques utilize confusion and diffusion operations within the frequency domain of an image. These techniques are computationally more expensive as compared to chaotic systems-based image encryption algorithms, and conceptually seem to provide a high security level. Nevertheless, the security analysis of frequency domain-based image encryption methods remains unexplored in literature.

Compressive sensing-based image encryption algorithms can reduce the size of image data and add an improved level of security to image transmission. These algorithms have a



disadvantage of high computational overhead resulting from combining compressive sensing and encryption operations. Furthermore, the effectiveness of compressive sensing depends on choosing the measurement matrix, which can affect the overall performance and efficiency of the algorithm.

Meaningful image encryption algorithms have the advantage of keeping the embedded cipher image secure even if it was extracted via steganalysis tools. Since the extracted data is already encrypted and is a noise like image, it adds another layer of security by preventing attackers from directly obtaining the secret image through steganalysis alone. However, these algorithms come with a drawback of high computational requirements due to the implementation of two layers of security.

The applications of image encryption are versatile, making it challenging to assign a specific technique to a particular use case. Image encryption techniques are relevant wherever image data protection and security are required. They are instrumental in personal communication, healthcare, finance, surveillance, law enforcement, and military applications. Nevertheless, certain image encryption techniques may excel for specific domains. Chaotic-based encryption algorithms are suitable for resource-constrained platforms, like the Internet of things (IoT) and edge devices<sup>64;65</sup>, due to their fast computational speed and minimal complexity. DNA-based algorithms can be utilized in encrypting and securely archiving data for extended periods, due to their minimal storage requirements. They also hold potential in espionage applications, as DNA sequences can be discreetly embedded in various forms, i.e. physical objects, and require specialized equipment for retrieval. Compressive sensing-based algorithms can be particularly useful in scenarios where there are constraints on data transmission bandwidth. For example, compressive sensing is well-suited for applications involving embedded cameras and drones, or any devices with limited data transfer capacity. Meaningful image encryption adds an additional layer of security through steganography, making it particularly suitable for covert communication.

Even though image encryption researchers utilize a comprehensive set of security, quality, and efficiency metrics to evaluate image encryption algorithms, there is a pressing need to establish rigorous security analysis and standardized evaluation metrics. Standardizing the

evaluation metrics will not only better assess the resilience of image encryption methods, but will also guide researchers in testing their proposed algorithms. Researches also need to consider robustness of the image encryption algorithms against novel attacks, such as adversarial machine learning and quantum computing.

Table 1.6: Strengths and weaknesses of different image encryption approaches.

| Approach              | Strengths                                                   | Weaknesses                                                                                                            |
|-----------------------|-------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------|
| Traditional           | Well established;<br>Secure;<br>Standardized and tested     | High computational requirement;<br>Slow encryption speed;<br>Faulty decryption of cipher images with corrupted pixels |
| Chaotic systems       | Fast computation;<br>Minimal complexity                     | Small range of chaotic behaviour                                                                                      |
| DNA                   | High security;<br>Less storage requirements                 | Experimental and only explored in academic research;<br>Require specialized lab equipment                             |
| Neural networks       | Fast encryption                                             | Security level not explored                                                                                           |
| Frequency domain      | Intuitively secure                                          | Computationally intensive;<br>Security analysis has not been explored                                                 |
| Compressive sensing   | High security;<br>Small size of cipher image                | High computational overhead                                                                                           |
| Meaningful encryption | High security due to combining encryption and steganography | High computational overhead                                                                                           |

## 1.5 Conclusion

The aim of this chapter is to provide researchers and practitioners new to the field of image encryption with an introduction to image encryption, and help enable them to grasp the current state of image encryption techniques and evaluation metrics. In this chapter, a comprehensive review of several cutting-edge image encryption algorithms is conducted, focusing on their respective strengths and weaknesses. We have categorized image encryption algorithms into seven distinct classifications based on the techniques they employ, that is, chaotic

systems, DNA encoding, neural networks, compressive sensing, frequency domain, and meaningful sensing. Furthermore, the chapter explored the security, quality, and efficiency metrics used in literature to evaluate image encryption algorithms, such as correlation coefficient, histogram analysis, entropy, gray-level co-occurrence matrix (GLCM) analysis, mean square error (MSE), peak signal-to-noise ratio (PSNR), unified average change intensity (UACI), number of pixel change rate (NPCR), NIST SP 800-22 test suite, and more.

# Chapter 2

## Image Encryption Based on Chaos

### 2.1 Introduction

Information security has an important role when it comes to sharing data. A lot of cryptography algorithms have been proposed for secure storage of information on computer systems as well as secure transfer of information over a network. Digital images are one form of sensitive data that needs to be stored and transmitted securely<sup>66</sup>. Digital images are two-dimensional arrays with different number of channels (one for grayscale images, three for color images, and four for color images with a transparency channel) that store pixel values. Digital images tend to have a high redundancy due to correlation between neighboring pixels.

Image encryption relies on two techniques to create cipher images and reduce the correlation between neighboring pixels: confusion and diffusion<sup>3</sup>. Confusion is often achieved via substitution, which is the change of the values of each pixel in a digital image by a substitution map. In image encryption, confusion is controlled by using a key to obscure the plaintext image values<sup>67</sup>. Diffusion means that if a single pixel is changed in the plaintext image, then it should result in a change in about half of the pixels in the cipher image, and similarly if a single pixel is changed in the cipher image, then it should result in a change in about half of the pixels in the plaintext image. Diffusion helps reduce the correlation between adjacent pixels in a plaintext image, which is accomplished by permutation in image

encryption algorithms.

Research in image encryption relies on a set of parameters to evaluate the security and efficiency of an image encryption method<sup>68</sup>. Some of the parameters are used to quantify the diffusion characteristics of an image encryption method, such as mean square error (MSE), mean absolute error (MAE), number of pixel change rate (NPCR), and unified average change intensity (UACI). To measure the quality of the encryption, metrics such as chi-square, peak signal-to-noise ratio (PSNR), maximum deviation, irregular deviation, deviation from uniform histogram, and resistance to noise and data loss attacks are used. Correlation coefficient, and local and global entropy analysis are parameters that assess the security of the encryption method. When deciding on the specific order of confusion and diffusion steps while designing an encryption algorithm, it needs to be ensured that the steps are reversible so that the cipher image can be decrypted. Furthermore, computational time and energy consumption of the image encryption algorithm are additional factors to consider depending on the application domain.

Most of the prior works on image encryption<sup>69–72</sup> do not consider encryption speed in their design. In this chapter, we propose a lightweight image encryption algorithm considering the encryption speed. In our proposed lightweight image encryption algorithm, chaotic maps are selected for their minimal computation requirement, low complexity, and fast speed in generating pseudorandom sequences. The algorithm utilizes a pre-shared symmetric key  $k_{sym}$  and an initialization vector (IV) to randomly generate two 2D matrices that are used to perform the confusion and diffusion on the plaintext image. The pre-shared symmetric key  $k_{sym}$  is XORed with hash of the plaintext image to obtain a derived key. The IV is a generated random data block for each run of the algorithm so that encryption of the same plaintext image yields a totally different cipher image for each run of the algorithm.

Our main contributions in this article are as follows:

- Developing a lightweight image encryption algorithm without compromising the security, quality, and efficiency metrics.
- Evaluating the proposed image encryption algorithm with a comprehensive set of eval-

uation metrics, such as correlation coefficient, histogram and chi-square tests, local and global entropy analysis, encryption quality, diffusion characteristics, resistance to differential attacks, resistance to noise and cropping attacks, and key sensitivity. The evaluation results of our proposed algorithm are compared with existing image encryption algorithms.

- Utilizing an IV with the proposed algorithm so that the encryption of the same plaintext image with the proposed algorithm yields totally different cipher images in each run of the algorithm, which helps in hiding statistical pattern in image encryption algorithms thus making the cryptanalysis difficult.
- Testing the randomness of the cipher images produced by the proposed image encryption algorithm using the National Institute of Standards and Technology (NIST) test suite.

The rest of the chapter is organized as follows. Section 2.2 presents the proposed image encryption algorithm along with its flowchart and pseudocode. Section 2.3 discusses the detailed evaluation and results of the proposed algorithm. Furthermore, The evaluation results are compared with existing image encryption algorithms. Finally, Section 2.4 concludes this chapter.

## 2.2 Proposed Image Encryption Algorithm

The flowchart for the proposed lightweight image encryption algorithm is shown in Fig. 2.1. The algorithm is designed to be lightweight and performs faster than other image encryption methods in literature without compromising the encryption quality and security metrics. The proposed method utilizes a logistic map to perform row and column permutation, and the AES S-Box and an XOR operation to perform the substitution. The AES S-Box is a non-linear substitution table that maps an 8-bit input to an 8-bit output<sup>4</sup>. The logistic map is used to generate two matrices that are utilized to perform confusion and diffusion on the

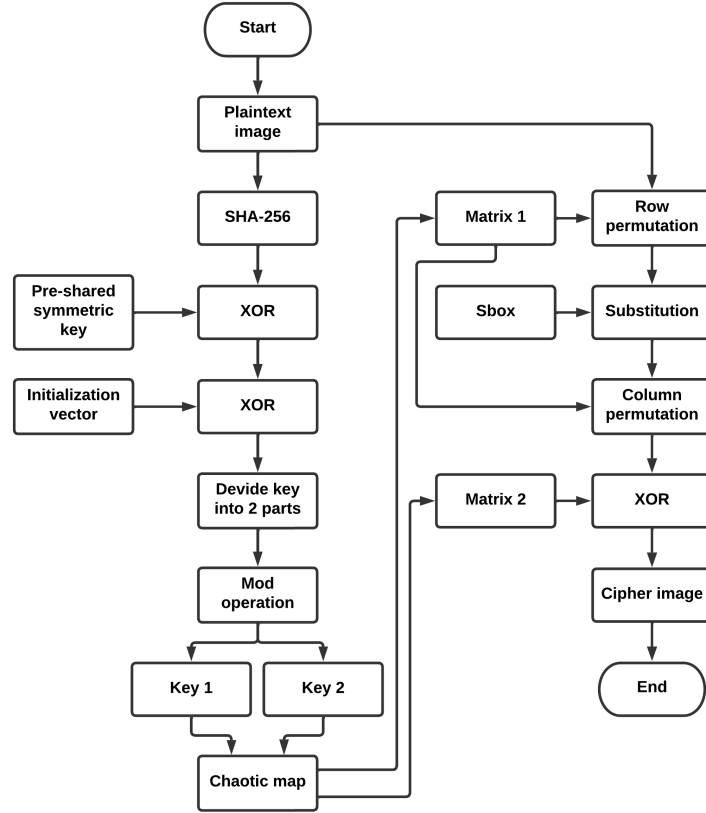


Figure 2.1: Proposed image encryption algorithm's flowchart.

plaintext image. The logistic map is given by the following recursive equation:

$$X_{n+1} = rX_n(1 - X_n), \quad (2.1)$$

where  $X_0$  is the starting population or initial parameter,  $X$  range is between  $[0, 1]$ ,  $r$  is the growth rate or control parameter, which has a range between  $[0, 4]$ , but the map only start to behave chaotically when  $r$  is between  $[3.56995, 4]$ <sup>73</sup>. Figure 2.2 depicts the bifurcation diagram of the logistic map. The S-Box and XOR operation are used to obscure the plaintext image values and reduce the correlation between the image pixels. The proposed algorithm comprises of two main functions. The first function generates the encryption keys matrices, using a combination of a hash generated from the plain text image using SHA-2 256, a pre-shared secret key, and an IV that is generated randomly for each run of the algorithm. SHA-2 256 is a hashing function that was developed by the United States National Security Agency

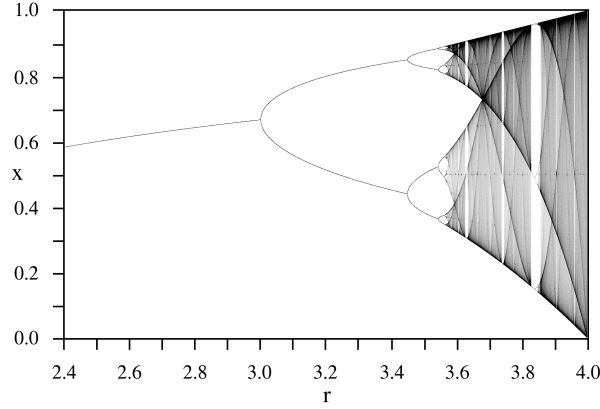


Figure 2.2: Logistic map bifurcation diagram.

(NSA). Hashing is a mathematical function that transforms data of any size into a bit array with a fixed size<sup>74</sup>. Even though SHA-3 is newer and more secure than SHA-2 256, we have not used SHA-3 in the proposed algorithm because of higher computational complexity of SHA-3 as compared to SHA-2. The hash generated from the plaintext image, pre-shared secret key, and IV are XORed to generate a 256-bit derived key, which is then split into two 128-bit parts ( $Key_1$ ,  $Key_2$ ) that goes through a `mod 0.9999` operation to be suitable for use as an initial parameter for the logistic map. The main purpose of the IV is that the same plaintext image produces totally different cipher images on multiple runs of the algorithm even though the pre-shared key has not changed. Thus, IV helps in hiding statistical pattern in image encryption, which makes the cryptanalysis difficult.  $Key_1$  and  $Key_2$  are used to generate Matrix 1 ( $M_1$ ) and Matrix 2 ( $M_2$ ), respectively. The second function of the proposed algorithm is the confusion and diffusion transformation to encrypt the plaintext image.  $M_1$  is used to chaotically permute the rows and the columns of the plaintext image. AES S-Box is used to obscure the plaintext image values, and  $M_2$  is used in an XOR operation on the resulting permuted and substituted image. Algorithm 1 depicts the pseudocode of the

Table 2.1: An example of the generation of Matrix 1 ( $M_1$ ) from the logistic map.

|     |     |     |     |     |
|-----|-----|-----|-----|-----|
| 22  | 140 | 246 | 206 | 238 |
| 76  | 186 | 252 | 20  | 134 |
| 78  | 251 | 21  | 164 | 214 |
| 82  | 68  | 86  | 180 | 12  |
| 166 | 226 | 136 | 215 | 144 |



proposed image encryption algorithm. Table 2.1 shows a sample of the generated  $M_1$  values. To illustrate the generation of the matrices, let's assume that  $Key_1$  initial value is 0.78648 (line 5 in Algorithm 1). The first iteration of the matrix generation uses the initial  $Key_1$  value to calculate the new  $Key_1$  using the logistic map (Eq. 2.1). The new  $Key_1$  value comes out to be  $3.99876 \times 0.78648 \times (1 - 0.78648) = 0.67151$ . The first value of  $M_1$  is calculated using the new  $Key_1$  (line 10 in Algorithm 1) as  $0.67151 \times 10^6 \bmod 256 = 671,510 \bmod 256 = 22$ . It is to be noted that we round the results to five significant figures for the fractional part of numbers in our computations for  $M_1$  and  $M_2$ . In the second iteration,  $Key_1$  from the previous iteration is used in the logistic map, and the new  $Key_1$  value comes out to be  $3.99876 \times 0.67151 \times (1 - 0.67151) = 0.88206$ . The second value of  $M_1$  is calculated using the new  $Key_1$  as  $0.88206 \times 10^6 \bmod 256 = 882,060 \bmod 256 = 140$ . Similarly, other values of  $M_1$  are calculated following the steps in Algorithm 1.

---

**Algorithm 1:** Proposed Lightweight Image Encryption

---

**Input:** Plaintext image  $P$ , pre-shared key  $k_{sym}$ , initialization vector  $IV$   
**Output:** Cipher image  $C$

```

1  $[H, W, nc] \leftarrow size(P)$  ; //Image height  $H$ , width  $W$ , and number of channels
    $nc$ 
2  $Hash \leftarrow hash(P)$  ; //Generates a hash from  $P$  using SHA256-2
3  $Key \leftarrow Hash \oplus k_{sym}$ ;
4  $Key \leftarrow Key \oplus IV$ ;
5  $Key_1 \leftarrow Key(0 : 127) \bmod m$  ; //m is set to 0.9999 in our implementation
6  $Key_2 \leftarrow Key(128 : 255) \bmod m$ ;
7 for  $i \leftarrow 1$  to  $H$ ,  $j \leftarrow 1$  to  $W$  do
8    $Key_1 \leftarrow r \times Key_1 \times (1 - Key_1)$  ; //from Eq. 2.1,  $r$  is set to 3.99876
9    $Key_2 \leftarrow r \times Key_2 \times (1 - Key_2)$  ; //from Eq. 2.1,  $r$  is set to 3.99876
10   $M_1(i, j) \leftarrow Key_1 \times 10^6 \bmod H$  ; //M1 denotes Matrix 1;  $H$  denotes image
    height
11   $M_2(i, j) \leftarrow Key_2 \times 10^6 \bmod 256$  ; //M2 denotes Matrix 2
12 end
13  $Q \leftarrow P$  ; //Q denotes the current state of the image
14  $Q \leftarrow rowPermutation(Q, M_1)$ ;
15  $Q \leftarrow SBox(Q)$ ;
16  $Q \leftarrow colPermutation(Q, M_1)$ ;
17  $Q \leftarrow Q \oplus M_2$ ;
18  $C \leftarrow Q$ ;
return:  $C$ 
```

---

The steps of the proposed algorithm to encrypt an image are as follows:

1. Read plaintext image  $P$ .
2. Generate a hash from plaintext image  $P$  using SHA-2 256 (line 2 in Algorithm 1).
3. Generate the IV random data block.
4. Perform XOR operation between the image hash and the pre-shared secret key  $k_{sym}$  (line 3 in Algorithm 1).
5. Perform XOR operation between the result of Step 4 and the IV (line 4 in Algorithm 1).
6. Divide the result of Step 5 into two equal parts of 128-bit.
7. Map the two parts from Step 6 between 0–0.9999 (by converting the hexadecimal hash part into an integer and then taking modulus 0.9999) and save as keys  $Key_1$  and  $Key_2$  (lines 5 and 6 in Algorithm 1).
8. Use the two keys as an initial parameter for the chaotic map to generate Matrix 1 ( $M_1$ ) and Matrix 2 ( $M_2$ ) (lines 7–11 in Algorithm 1).
9. Use  $M_1$  to perform chaotic row permutation on  $P$  (line 14 in Algorithm 1).
10. Use the AES S-Box to replace the pixel values of the resulting permuted image from Step 9 (line 15 in Algorithm 1).
11. Use  $M_1$  to perform chaotic column permutation on the resulting substituted image from Step 10 (line 16 in Algorithm 1).
12. Perform XOR operation between the resulting permuted image from Step 11 and Matrix 2 ( $M_2$ ) to produce the cipher image  $C$  (lines 17 and 18 in Algorithm 1).

To illustrate these steps further, the proposed algorithm is applied on a  $4 \times 4$  sample data and presented in Fig. 2.3. For decryption, cipher image  $C$ , IV, and the hash of the image are sent to the receiver. The SHA-2 256 hash is securely sent to the receiver along with the

key via a secure key exchange algorithm, such as Diffie–Hellman key exchange. Steps of the decryption process for the proposed algorithm are discussed below:

1. Read the cipher image  $C$  along with the SHA-2 256 hash and the IV.
2. Perform Steps 4 to Steps 8 of the encryption process.
3. Perform XOR operation between the cipher image  $C$  and  $M_2$ .
4. Use  $M_1$  to perform chaotic column permutation on the resulting XORed image from Step 4.
5. Use the AES reverse S-Box to replace the pixel values of the resulting permuted image from Step 5.
6. Use  $M_1$  to perform chaotic row permutation on the resulting substituted image from Step 6 to produce the decrypted image  $I$ .



Figure 2.3: The proposed algorithm applied to a  $4 \times 4$  sample.

## 2.3 Results and Analysis

To evaluate the proposed algorithm, we have used a set of standard test images found in the literature<sup>75</sup>, namely Baboon, Peppers, Male, Sailboat, and Cameraman, as shown in Fig. 2.4. The proposed algorithm is tested on 8-bit grayscale images, and 24-bit color images with variable resolutions of  $128 \times 128$ ,  $256 \times 256$ ,  $512 \times 512$ , and  $1024 \times 1024$  pixels. A set of plaintext images and their corresponding cipher images are shown in Fig. 2.5. The proposed algorithm is analyzed and compared to methods from related literature for time complexity, correlation coefficient, analyses of histogram, chi-square, local and global entropy, encryption quality (MSE, MAE, PSNR, maximum deviation, irregular deviation, and deviation from

uniform histogram), resistance to differential attacks, NPCR, UACI, resistance to noise and data loss attacks, homogeneity, contrast, energy, key space, and key sensitivity. The NIST SP 800-22 test suite is also used to test the randomness characteristics of the proposed algorithm. Each of these tests and their results are discussed in detail below.

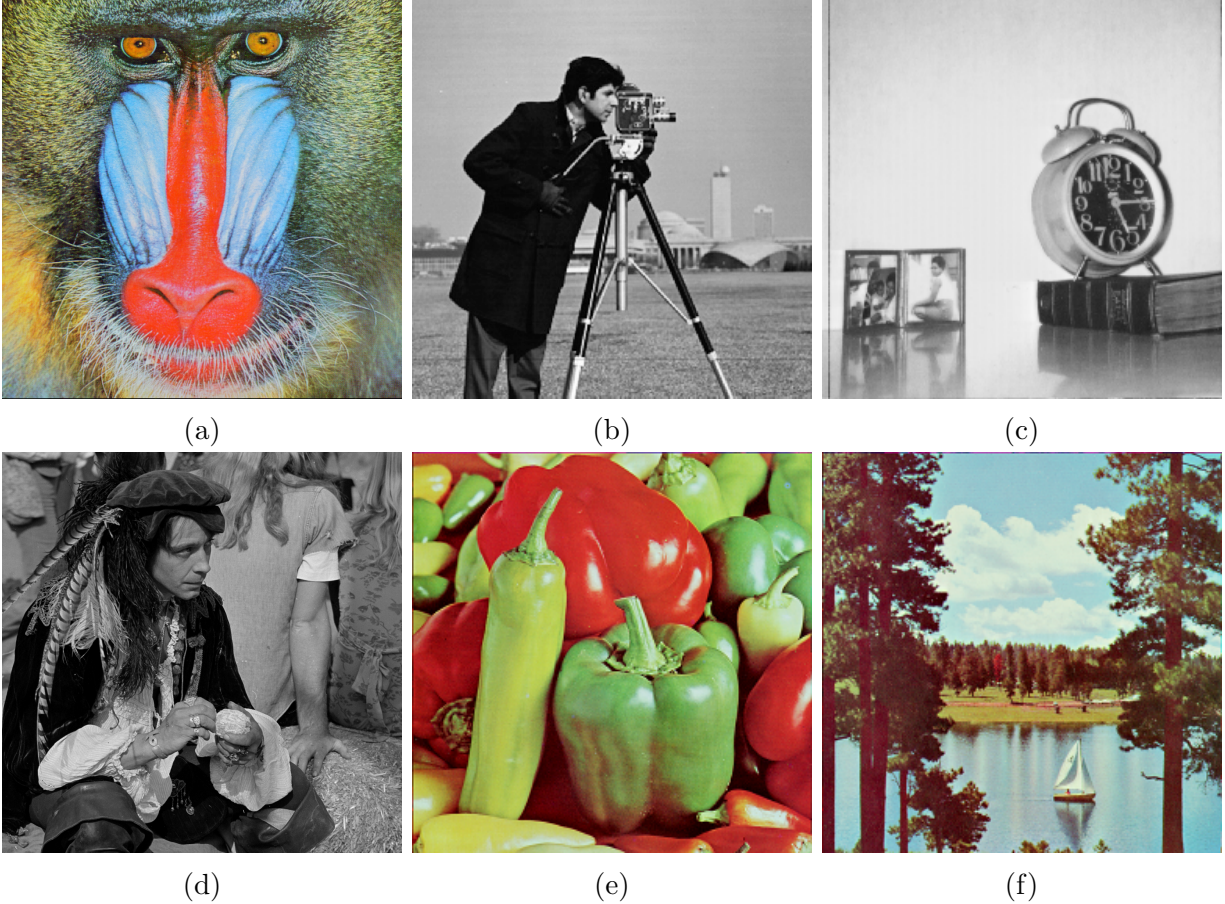


Figure 2.4: Images used to test the proposed algorithm: **(a)** Baboon, **(b)** Cameraman, **(c)** Clock, **(d)** Male, **(e)** Peppers, **(f)** Sailboat.

### 2.3.1 Encryption Speed

The machine used to test the performance of the proposed algorithm has an Intel Core i5 CPU running at 3.5GHz. The machine has a 12GB of RAM and runs Windows 10 operating system. The proposed algorithm is run in MATLAB R2022b. Table 2.2 shows the results of execution time of the algorithm on images with sizes  $256 \times 256$ ,  $512 \times 512$ , and  $1024 \times 1024$ . Table 2.3 compares the encryption speed of the proposed algorithm with

the speeds of several related image encryption methods in the literature. Results indicate that the proposed algorithm has the fastest encryption speed. In Table 2.4, the execution times of the related encryption methods have been scaled to 3.5 GHz CPU to match with the processor frequency of the machine on which the proposed algorithm is run. Even though scaling does not provide 100% accuracy for processor runtime because of different instruction set architectures and memory subsystem, scaling provides reasonable estimates and facilitates relative comparisons<sup>76</sup>. Results indicate that the proposed algorithm still has the best encryption speed after scaling. The proposed algorithm has a speed up of 2.36x compared to<sup>70</sup>, 15.30x speed up compared to<sup>77</sup>, and a speed up of 2.29x when compared to<sup>53</sup>.

Table 2.2: Average encryption time (in seconds) of the proposed algorithm for different images dimensions.

| Image size         | Color | Time (sec) |
|--------------------|-------|------------|
| $256 \times 256$   | Gray  | 0.0235     |
| $256 \times 256$   | Color | 0.0287     |
| $512 \times 512$   | Gray  | 0.0915     |
| $512 \times 512$   | Color | 0.1056     |
| $1024 \times 1024$ | Gray  | 0.4124     |
| $1024 \times 1024$ | Color | 0.4386     |

Table 2.3: Average encryption time (in seconds) of several color and gray scale  $512 \times 512$  images for the proposed algorithm compared to different related methods.

|            | Proposed       | <sup>69</sup> | <sup>70</sup> | <sup>71</sup> | <sup>77</sup> |
|------------|----------------|---------------|---------------|---------------|---------------|
| Time (sec) | <b>0.09155</b> | 0.2338        | 0.3033        | 1.28          | 1.489         |

Table 2.4: Encryption time (in seconds) for the proposed algorithm compared to different related methods scaled to 3.5GHz CPU.

|            | Proposed       | <sup>70</sup> | <sup>77</sup> | <sup>53</sup> |
|------------|----------------|---------------|---------------|---------------|
| Time (sec) | <b>0.09155</b> | 0.2166        | 1.4039        | 0.2102        |
| Speed up   | -              | 2.36          | 15.33         | 2.30          |



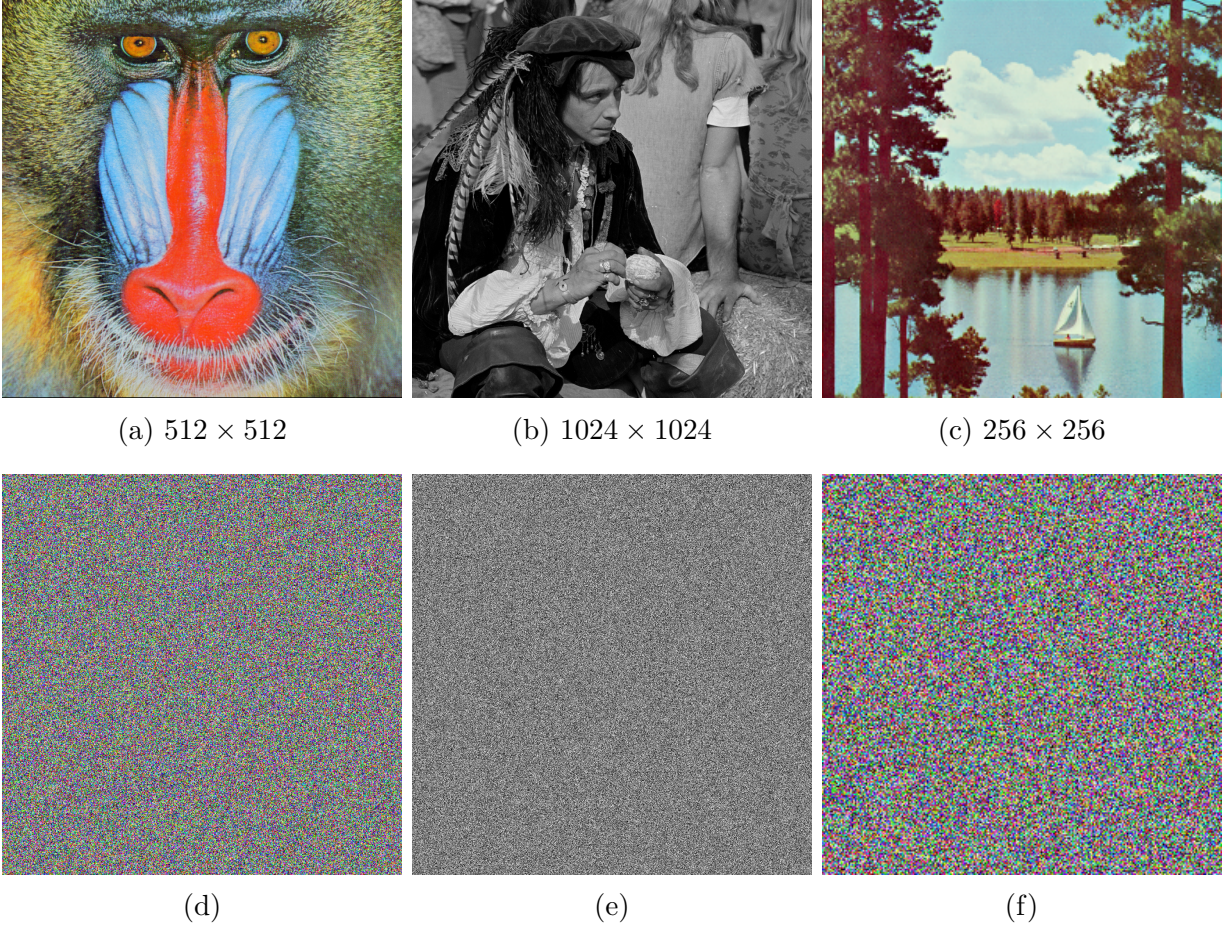


Figure 2.5: Sample images and their respective encrypted ciphers: (a) Baboon, (b) Male, (c) Sailboat, (d) Baboon cipher, (e) Male cipher, (f) Sailboat cipher.

### 2.3.2 Correlation Coefficient Analysis

Images tend to have a high correlation between neighboring pixels. Image encryption methods aim to reduce this correlation to help obscure the image. The correlation values range between  $[-1, 1]$ , where 0 means no correlation between the two pixels, and 1, -1 represent the maximum positive and negative correlation, respectively. For an image encryption algorithm to be considered secure, the correlation values should be as close to 0 as possible. To test the correlation of the pixels in the cipher image two methods are used: (i) correlation coefficient between adjacent pixels of the cipher image<sup>78</sup>, and (ii) correlation coefficient between the plaintext image and its cipher<sup>79</sup>. The correlation coefficient is calculated and compared for the horizontal, vertical, and diagonal pixels of the image.

## Correlation Coefficient of Adjacent Pixels

The results for the vertical, horizontal, and diagonal correlation coefficients of adjacent pixels for the cipher images for the proposed algorithm are presented in Table 2.5. Table 2.6 compares vertical, horizontal, and diagonal correlation coefficients for the proposed algorithm to related encryption methods in the literature. The results show that the proposed algorithm has smaller values of correlation coefficients compared to other algorithms, but the performance of encryption algorithm by Alanezi et al.<sup>70</sup> is better than the proposed algorithm for this metric though the results of the proposed algorithm are still very close to that of the encryption algorithm by Alanezi et al.<sup>70</sup>.

Table 2.5: Vertical, horizontal, and diagonal correlation coefficients of adjacent pixels of encrypted images.

| Image     | Color and size   | Correlation coefficient |            |          |
|-----------|------------------|-------------------------|------------|----------|
|           |                  | Vertical                | Horizontal | Diagonal |
| Baboon    | 512 × 512 Color  | -0.0001                 | 0.0006     | -0.0021  |
| Cameraman | 512 × 512 Gray   | -0.0039                 | -0.0003    | 0.0037   |
| Male      | 1024 × 1024 Gray | 0.0004                  | 0.0017     | -0.0009  |
| Peppers   | 256 × 256 Color  | -0.0003                 | 0.0009     | -0.0006  |
| Sailboat  | 512 × 512 Color  | -0.0012                 | 0.0016     | 0.0001   |
| Average   |                  | 0.0005                  | 0.0004     | -0.0004  |

Table 2.6: Vertical, horizontal, and diagonal correlation coefficients of encrypted Baboon image of the proposed algorithm compared to different encryption methods.

|               | Vertical       | Correlation coefficient |               |
|---------------|----------------|-------------------------|---------------|
|               |                | Horizontal              | Diagonal      |
| Proposed      | <b>-0.0001</b> | 0.0006                  | -0.0021       |
| <sup>69</sup> | -0.0086        | 0.0023                  | 0.0402        |
| <sup>70</sup> | <b>-0.0001</b> | <b>-0.0002</b>          | <b>0.0011</b> |
| <sup>71</sup> | -0.0036        | -0.0019                 | -0.0033       |
| <sup>77</sup> | -0.0004        | 0.0007                  | 0.0029        |

### Correlation Coefficient Between Plaintext and Cipher Images

The vertical, horizontal, and diagonal correlation coefficients of the proposed algorithm are presented in Table 2.7. Results show that the proposed algorithm has small correlation values between the plaintext and their cipher images.

Table 2.7: Vertical, horizontal, and diagonal correlation coefficients of plaintext and cipher images.

| Image     | Color and size          | Correlation coefficient |            |          |
|-----------|-------------------------|-------------------------|------------|----------|
|           |                         | Vertical                | Horizontal | Diagonal |
| Baboon    | $512 \times 512$ Color  | -0.0029                 | -0.0550    | -0.0005  |
| Cameraman | $512 \times 512$ Gray   | -0.0912                 | -0.0353    | 0.0035   |
| Male      | $1024 \times 1024$ Gray | 0.0164                  | -0.0226    | -0.0007  |
| Peppers   | $256 \times 256$ Color  | -0.0025                 | 0.0023     | 0.0002   |
| Sailboat  | $512 \times 512$ Color  | -0.1277                 | 0.0265     | 0.0012   |
| Average   |                         | -0.0372                 | -0.0061    | 0.0003   |

### 2.3.3 Histogram Analysis

Table 2.8 shows the chi-square test values for plaintext and cipher images encrypted by the proposed algorithm. Results in Table 2.8 indicate that the  $\chi^2$  values of the cipher images encrypted by our proposed algorithm are extremely low, and thus the histograms of cipher images encrypted by our proposed encryption algorithm are very close to uniform distribution.

Table 2.8: Chi-square test values for plaintext images and cipher images encrypted by our proposed algorithm.

| Image     | Size and color          | Chi-square      |              |
|-----------|-------------------------|-----------------|--------------|
|           |                         | Plaintext Image | Cipher Image |
| Baboon    | $512 \times 512$ Color  | 10429131.335    | 258.671      |
| Cameraman | $512 \times 512$ Gray   | 2193251.085     | 247.173      |
| Male      | $1024 \times 1024$ Gray | 26095050.882    | 299.851      |
| Peppers   | $256 \times 256$ Color  | 557983.062      | 286.742      |
| Sailboat  | $512 \times 512$ Color  | 424683.429      | 273.918      |
| Average   |                         |                 | 273.271      |



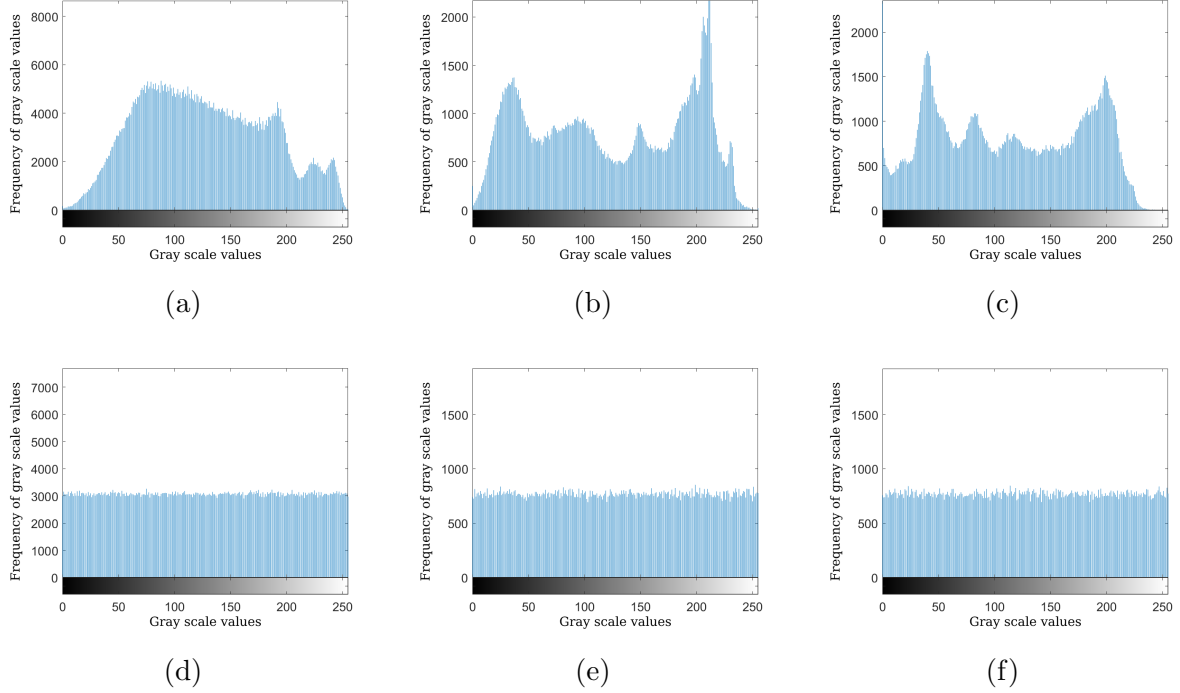


Figure 2.6: Histograms of images and their respective ciphers: (a) Baboon histogram, (b) Sailboat histogram, (c) Peppers histogram, (d) Baboon cipher histogram, (e) Sailboat cipher histogram, (f) Peppers cipher histogram.

### 2.3.4 Entropy Analysis

The results of the entropy analysis of the proposed algorithm are presented in Table 2.9. Table 2.10 compares the proposed algorithm global Shannon entropy with other encryption methods in the literature. The calculated entropy values of the proposed algorithm are closer to the ideal entropy and higher than those of the compared methods.

### 2.3.5 Encryption Quality

Encryption quality is an important factor in testing an image encryption method's efficiency. To quantify the quality of encryption, different tests are performed on the plaintext images and their respective cipher images, such as mean square error (MSE), PSNR, maximum deviation, irregular deviation, and deviation from the uniform histogram. Each of these tests and their results are discussed in detail below.

Table 2.9: Global and local Shannon entropy values of images encrypted by our proposed algorithm.

| Image     | Size and color          | Entropy |        |
|-----------|-------------------------|---------|--------|
|           |                         | Global  | Local  |
| Baboon    | $512 \times 512$ Color  | 7.9997  | 7.8979 |
| Cameraman | $512 \times 512$ Gray   | 7.9991  | 7.8978 |
| Male      | $1024 \times 1024$ Gray | 7.9996  | 7.9026 |
| Peppers   | $256 \times 256$ Color  | 7.9990  | 7.8883 |
| Sailboat  | $512 \times 512$ Color  | 7.9988  | 7.8880 |
| Average   |                         | 7.9992  | 7.8949 |

Table 2.10: Global entropy of cipher image Baboon of the proposed algorithm compared with different encryption algorithms.

|         | Proposed      | 69     | 70     | 71     | 77     |
|---------|---------------|--------|--------|--------|--------|
| Entropy | <b>7.9997</b> | 7.9024 | 7.9991 | 7.9992 | 7.9971 |

### Mean Square Error (MSE)

Table 2.11 lists the MSE values between the plaintext and cipher images for the proposed algorithm. Table 2.12 compares the proposed algorithm with other image encryption methods in the literature. The comparison results show that images encrypted by the proposed algorithm have better encryption quality with respect to the MSE metric as compared to other encryption methods.

Table 2.11: MSE values of images encrypted by the proposed algorithm.

| Image     | Size and color          | MSE     |
|-----------|-------------------------|---------|
| Baboon    | $512 \times 512$ Color  | 46.0304 |
| Cameraman | $512 \times 512$ Gray   | 49.8847 |
| Male      | $1024 \times 1024$ Gray | 53.8392 |
| Peppers   | $256 \times 256$ Color  | 23.5767 |
| Sailboat  | $512 \times 512$ Color  | 53.6684 |
| Average   |                         | 42.3794 |

Table 2.12: Mean Square Error (MSE) for images encrypted by the proposed algorithm compared to related methods.

|     | Proposed       | 71      | 80      | 81      |
|-----|----------------|---------|---------|---------|
| MSE | <b>42.3794</b> | 39.6794 | 33.4275 | 40.3295 |

### Mean Absolute Error (MAE)

Table 2.13 lists the MAE values for the proposed algorithm. Table 2.14 compares the proposed algorithm with other image encryption methods in the literature. The comparison results show that images encrypted by the proposed algorithm have better encryption quality with respect to the MAE metric than other encryption methods.

Table 2.13: MAE values of images encrypted by the proposed algorithm.

| Image     | Size and color          | MAE   |
|-----------|-------------------------|-------|
| Baboon    | $512 \times 512$ Color  | 89.69 |
| Cameraman | $512 \times 512$ Gray   | 96.56 |
| Male      | $1024 \times 1024$ Gray | 97.55 |
| Peppers   | $256 \times 256$ Color  | 86.47 |
| Sailboat  | $512 \times 512$ Color  | 87.64 |
| Average   |                         | 91.58 |

Table 2.14: Average MAE values for images encrypted by the proposed algorithm compared to related methods.

|     | Proposed     | 82    | 83    | 84 |
|-----|--------------|-------|-------|----|
| MAE | <b>91.58</b> | 79.57 | 78.10 | 90 |

### Peak Signal-to-Noise Ratio

Table 2.15 lists the PSNR values of encrypted images produced by the proposed algorithm, and Table 2.16 compares the PSNR values of the proposed algorithm to other image encryption methods in related literature. The comparison results show that images encrypted by the proposed algorithm have lower PSNR values and thus better encryption quality than

other encryption methods.

Table 2.15: Peak signal-to-noise ratio (PSNR) of images encrypted by the proposed algorithm.

| Image     | Size and color          | PSNR   |
|-----------|-------------------------|--------|
| Baboon    | $512 \times 512$ Color  | 8.7880 |
| Cameraman | $512 \times 512$ Gray   | 8.4124 |
| Male      | $1024 \times 1024$ Gray | 8.0008 |
| Peppers   | $256 \times 256$ Color  | 8.6210 |
| Sailboat  | $512 \times 512$ Color  | 8.7439 |
| Average   |                         | 8.4458 |

Table 2.16: Peak signal-to-noise ratio (PSNR) of the images encrypted by the proposed algorithm compared to other encryption methods.

|      | Proposed      | <a href="#">70</a> | <a href="#">71</a> | <a href="#">82</a> | <a href="#">53</a> |
|------|---------------|--------------------|--------------------|--------------------|--------------------|
| PSNR | <b>8.4458</b> | 8.6449             | 9.5424             | 9.0996             | 9.7936             |

## Maximum Deviation

The maximum deviation for images Baboon, Cameraman, and Peppers encrypted with the proposed algorithm are compared with other related encryption methods in Table [2.17](#). The results of the maximum deviation of the cipher images produced by the proposed algorithm are highly deviated from the original image. However, even though the performance of Arif et al. [71](#) is better than the proposed algorithm for some encrypted images, the average of the proposed method is still better in general.

Table 2.17: Maximum deviation results for proposed algorithm compared to related methods.

| Image     | proposed      | <a href="#">71</a> | <a href="#">85</a> |
|-----------|---------------|--------------------|--------------------|
| Baboon    | <b>363121</b> | 199158             | -                  |
| Cameraman | 64382         | <b>64998</b>       | 18007              |
| Peppers   | <b>209618</b> | 146408             | 22935              |
| Average   | <b>167482</b> | 102022             | 20109              |

## Irregular Deviation

The irregular deviation for images Baboon, Cameraman, and Peppers encrypted with the proposed algorithm are presented and compared to related methods in Table 2.18. Results indicate that the proposed algorithm has smaller irregular deviation values for most of the encrypted images as compared to other encryption methods. However, the performance of Belazi et al.<sup>85</sup> is better than the proposed algorithm for the image Peppers and on average.

Table 2.18: Irregular deviation results for the proposed algorithm compared to other encryption methods.

| Image     | Proposed     | 71     | 85           |
|-----------|--------------|--------|--------------|
| Baboon    | <b>59921</b> | 80203  | -            |
| Cameraman | <b>32165</b> | 32706  | 39244        |
| Peppers   | 76075        | 84465  | <b>35088</b> |
| Average   | 91460        | 129253 | <b>40739</b> |

## Deviation from Uniform Histogram

The deviation from uniform histogram for images Baboon, Cameraman, and Peppers encrypted with the proposed algorithm are presented in Table 2.19 and compared to other encryption methods. Results indicate that the deviation from uniform histogram for all of the cipher images produced by the proposed algorithm are lower than the compared methods.

Table 2.19: Deviation from uniform histogram for the proposed algorithm compared with other encryption methods.

| Image     | proposed      | 85     | 86      |
|-----------|---------------|--------|---------|
| Baboon    | <b>0.0256</b> | -      | 0.0496  |
| Cameraman | <b>0.0305</b> | 0.0942 | 0.0502  |
| Peppers   | <b>0.0315</b> | 0.0917 | -       |
| Average   | <b>0.0382</b> | 0.0534 | 0.04965 |

### 2.3.6 Resistance to Differential Attacks

Attackers utilize differential attacks (also known as chosen plaintext attacks) to guess the relation between the plaintext image and its cipher image to break the encryption algorithm<sup>68</sup>. In this method, the attacker encrypts two plaintext images with a 1-bit difference and compares the resulting cipher images. For an encryption algorithm to be considered secure, a pixel change in the plaintext image should yield a completely different cipher image. To evaluate an algorithm's resistance to differential attacks, a set of commonly used tests are performed. These tests are: the avalanche effect, number of pixels change rate (NPCR), and the unified averaged changing intensity (UACI). Each of these tests and their results are discussed in detail below.

#### Avalanche Effect

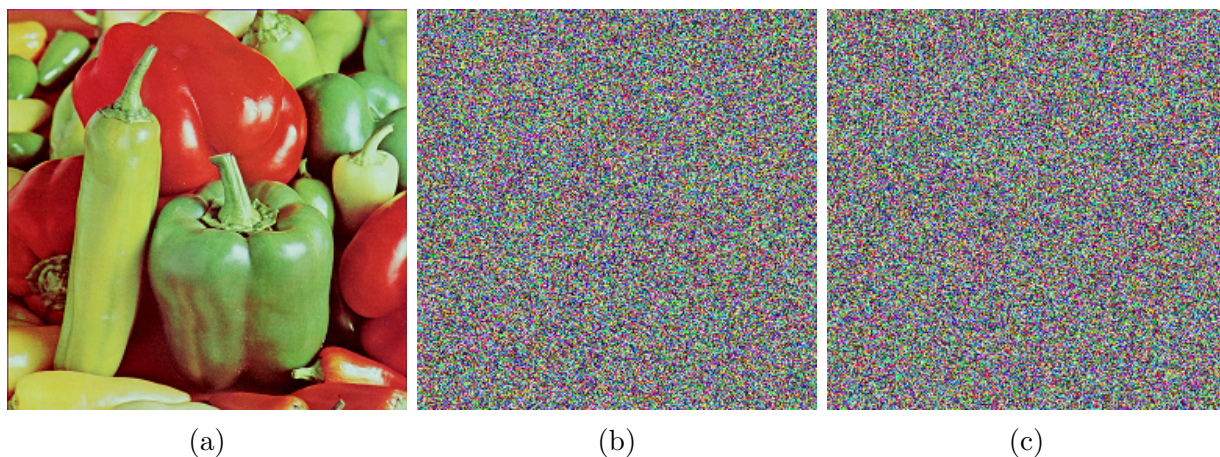


Figure 2.7: Avalanche effect illustrated in two images encrypted from the same plaintext image with a change of 1-bit, (d) Peppers, (e) Peppers cipher, (f) Peppers cipher where plaintext peppers have 1-bit change.

Fig. 2.7 depicts the plaintext image Peppers and the resulting ciphers with a change of 1-bit. If 1-bit change in the plaintext image causes a change of more than 50% in cipher image than the algorithm is safe against differential attacks<sup>55</sup>. Table 2.20 depicts  $MSE_{av}$  for various images encrypted by the proposed algorithm.

### Number of Pixels Change Rate (NPCR)

Table 2.20 depicts NPCR values for various images (Baboon, Cameraman, Male, Peppers, and Sailboat) encrypted by the proposed algorithm. The higher the NPCR value, the higher an algorithm's responsiveness to plaintext change is. The ideal average value of NPCR is  $\geq 99.6094$ <sup>57</sup>. Table 2.21 compares the average NPCR values of the proposed algorithm with other encryption algorithms discussed in the literature. Results shows that the proposed algorithm has higher average NPCR value as compared to other algorithm and the ideal value for NPCR. However, the performance of algorithm proposed by Alanezi et al.<sup>70</sup> is slightly better than the proposed algorithm with respect to NPCR metric.

### Unified Average Changing Intensity (UACI)

Table 2.20 depicts UACI values for various images (Baboon, Cameraman, Male, Peppers, and Sailboat) encrypted by the proposed algorithm. The higher the UACI values, the higher the algorithm responsiveness to plaintext change is. The ideal average value of UACI is  $\geq 33.4635$ <sup>57</sup>. Table 2.21 compares UACI of the proposed algorithm with other encryption algorithms. Results shows that the proposed algorithm has higher average UACI value as compared to other algorithms and the ideal value. However, the performance of the encryption algorithm proposed by Alanezi et al.<sup>70</sup> is slightly better than the proposed algorithm with respect to UACI metric.

Table 2.20: Avalanche effect  $MSE_{av}$ , NPCR, and UACI values of images encrypted by the proposed algorithm.

| Image     | Size and color          | $MSE_{av}$ | NPCR    | UACI    |
|-----------|-------------------------|------------|---------|---------|
| Baboon    | $512 \times 512$ Color  | 57.3895    | 99.6125 | 33.5531 |
| Cameraman | $512 \times 512$ Gray   | 57.3470    | 99.6368 | 33.3911 |
| Male      | $1024 \times 1024$ Gray | 57.2375    | 99.6146 | 33.4947 |
| Peppers   | $256 \times 256$ Color  | 57.4122    | 99.6218 | 33.4862 |
| Sailboat  | $512 \times 512$ Color  | 57.4562    | 99.6176 | 33.4993 |
| Average   |                         | 57.3533    | 99.6153 | 33.4718 |

Table 2.21: NPCR and UACI of the proposed algorithm compared with other encryption methods.

|          | NPCR           | UACI           |
|----------|----------------|----------------|
| Proposed | 99.6153        | 33.4718        |
| 69       | 99.5893        | 33.3730        |
| 70       | <b>99.6239</b> | <b>33.5615</b> |
| 71       | 99.6059        | 33.4375        |
| 77       | 99.5743        | 33.3941        |
| 87       | 99.6146        | 33.4501        |

### 2.3.7 Resistance to Noise and Data Loss Attacks

Digital images transferred over a network are prone to noise and data loss that could affect the encrypted image. Data loss attacks are sometimes also referred to as occlusion attacks. An *occlusion attack* is a cyberattack where an adversary with the capability to alter cipher image data can crop part of the image out to thwart decryption or to render the decrypted image useless. An image encryption algorithm should be robust against noise and data loss attacks and should be able to decrypt the affected cipher image. We have evaluated the robustness of the proposed algorithm against the data loss attacks by cropping parts of the cipher image. To measure the robustness against noise, we have added salt and pepper noise in the cipher image. Fig. 2.8 depicts the data loss (occlusion) attacks on cipher image with different severities (10% cropping, 25% cropping, and 50% cropping). Fig. 2.8 also illustrates that the plaintext image is successfully recovered by decrypting the affected cipher image with cropping up to 50% using our proposed encryption/decryption algorithm. Fig. 2.9 shows cipher images in which salt and pepper noise has been introduced with different densities (0.15, 0.25, and 0.5). We note that a noise density of 0.15 affects approximately 15% of pixels, a noise density of 0.25 affects approximately 25% of pixels, and a noise density of 0.5 affects approximately 50% of pixels in the image. Fig. 2.9 shows that our proposed algorithm is able to decrypt cipher images affected by salt and pepper noise density up to 0.5.



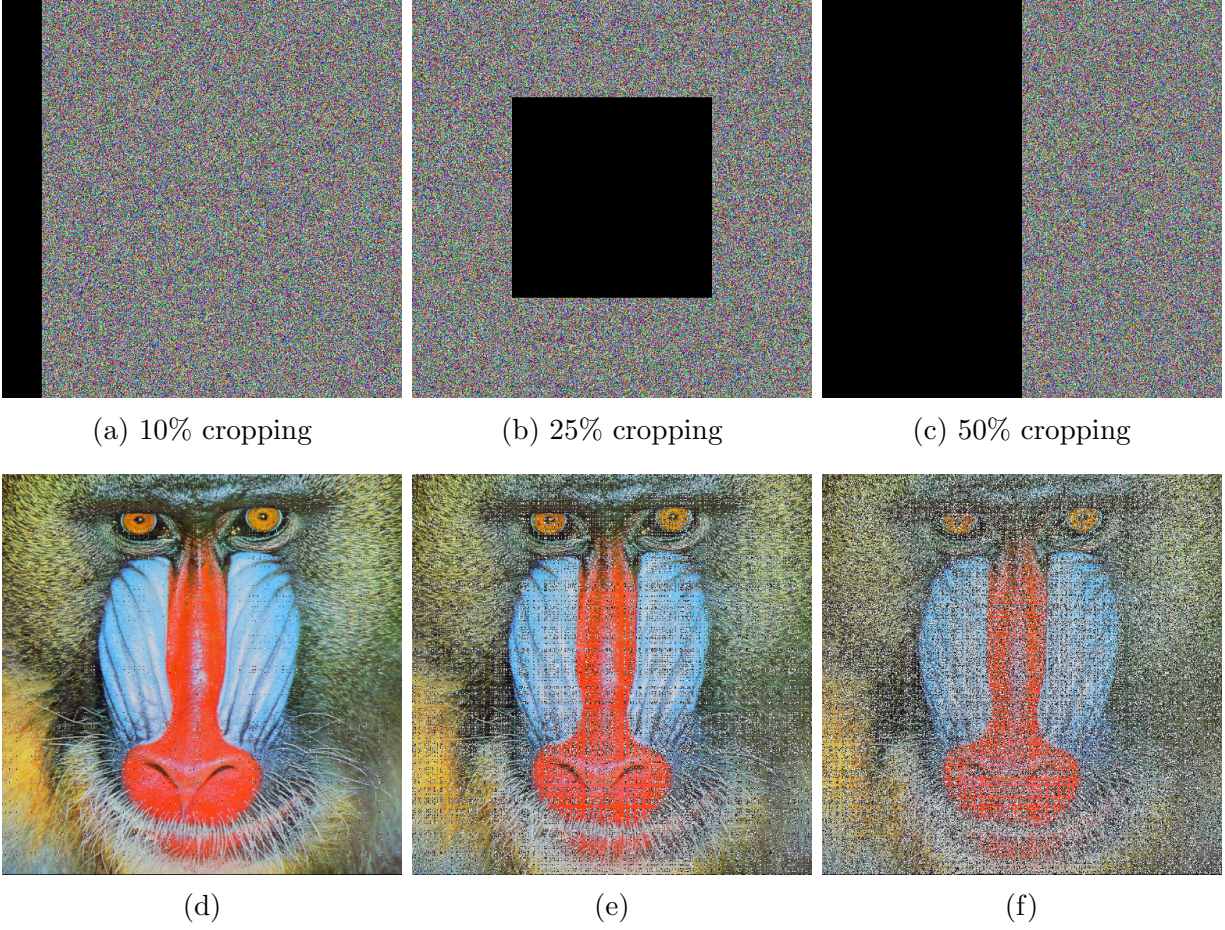


Figure 2.8: Results of data loss (occlusion) attacks on the color image Baboon ( $512 \times 512$ ): (a)-(c) different severities of occlusion attacks, (d)-(f) corresponding decrypted images.

### 2.3.8 Homogeneity

Table 2.22 lists the results of the homogeneity analysis of plaintext and cipher images. These results show that the proposed algorithm produces low values of homogeneity. Table 2.25 compares the homogeneity results of the proposed algorithm and other image encryption methods. Results in Table 2.25 indicate that our proposed algorithm produces the lowest homogeneity values when compared to other encryption methods.

### 2.3.9 Contrast

Table 2.23 shows the results of the contrast analysis of plaintext and cipher images for the proposed algorithm. These results show that the proposed algorithm is producing high



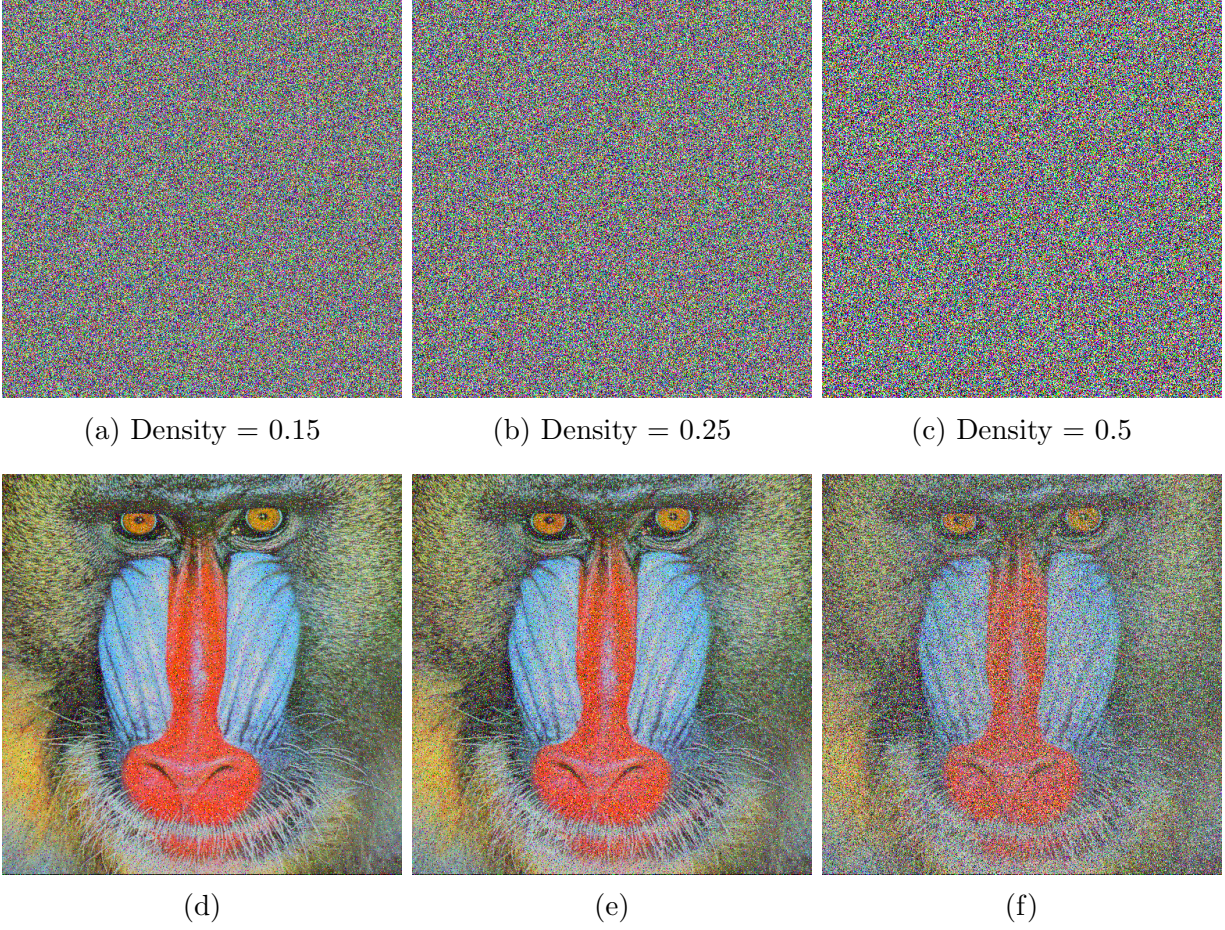


Figure 2.9: Results of salt and pepper noise attack on the color image Baboon ( $512 \times 512$ ): (a)-(c) different noise densities introduced, (d)-(f) corresponding decrypted images.

contrast levels in the encrypted image. Table 2.25 compares the contrast value produced by the proposed algorithm versus other image encryption methods. Results in Table 2.25 indicate that our proposed algorithm produces higher contrast value as compared to other encryption methods.

### 2.3.10 Energy

Table 2.24 shows the results of the energy analysis of plaintext and cipher images for our proposed algorithm. Table 2.25 compares the results of the energy analysis of cipher images produced by the proposed algorithm versus other encryption methods. Results in Table 2.25 show that the energy value of the cipher image from our proposed algorithm is among the

Table 2.22: Homogeneity analysis for plaintext and cipher images.

| Image     | Size and Color          | Plaintext Image | Cipher Image |
|-----------|-------------------------|-----------------|--------------|
| Baboon    | $512 \times 512$ Color  | 0.7652          | 0.3893       |
| Cameraman | $512 \times 512$ Gray   | 0.8979          | 0.3898       |
| Male      | $1024 \times 1024$ Gray | 0.8475          | 0.3897       |
| Peppers   | $256 \times 256$ Color  | 0.9353          | 0.3896       |
| Sailboat  | $512 \times 512$ Color  | 0.8664          | 0.3897       |
| Average   |                         |                 | 0.3897       |

Table 2.23: Contrast analysis for plaintext and cipher images.

| Image     | Size and Color          | Plaintext Image | Cipher Image |
|-----------|-------------------------|-----------------|--------------|
| Baboon    | $512 \times 512$ Color  | 0.7425          | 10.5152      |
| Cameraman | $512 \times 512$ Gray   | 0.1978          | 10.4670      |
| Male      | $1024 \times 1024$ Gray | 0.2527          | 10.4918      |
| Peppers   | $256 \times 256$ Color  | 0.2665          | 10.4893      |
| Sailboat  | $512 \times 512$ Color  | 0.4140          | 10.4868      |
| Average   |                         |                 | 10.5081      |

algorithms with lowest energy values.

Table 2.24: Energy analysis for plaintext and cipher images.

| Image     | Size and Color          | Plaintext Image | Cipher Image |
|-----------|-------------------------|-----------------|--------------|
| Baboon    | $512 \times 512$ Color  | 0.0639          | 0.01564      |
| Cameraman | $512 \times 512$ Gray   | 0.1939          | 0.01565      |
| Male      | $1024 \times 1024$ Gray | 0.1199          | 0.01563      |
| Peppers   | $256 \times 256$ Color  | 0.1437          | 0.01563      |
| Sailboat  | $256 \times 256$ Color  | 0.1155          | 0.01563      |
| Average   |                         |                 | 0.01563      |

### 2.3.11 Key Space and Sensitivity Analyses

The key space is the set of all possible keys that could be used by a cryptosystem. To provide sufficient security against brute force attacks, key space for an encryption algorithm should be larger than  $2^{100}$ <sup>60</sup>. The proposed algorithm relies on a 256-bit hash generated from the plaintext image using SHA-2 256, a 256-bit pre-shared secret key, and a 256-bit IV that are

Table 2.25: Comparison of homogeneity, contrast, and energy analysis of cipher image of the proposed algorithm and related methods.

|                    | Image Color | Homogeneity   | Contrast       | Energy         |
|--------------------|-------------|---------------|----------------|----------------|
| Proposed           | Color       | <b>0.3837</b> | <b>10.5081</b> | 0.01563        |
| <a href="#">82</a> | Color       | 0.3895        | 10.5079        | 0.01562        |
| <a href="#">88</a> | Gray scale  | 0.4208        | 8.3301         | 0.01760        |
| <a href="#">89</a> | Color       | 0.4110        | 8.6448         | <b>0.01561</b> |
| <a href="#">90</a> | Gray scale  | 0.3916        | 10.4252        | 0.01563        |

XORed together to produce a derived security key with a length of 256-bit, which is supplied as input parameter to the logistic map (Algorithm 1). Thus, the key space for the proposed algorithm is  $2^{256}$ , which is large enough to withstand brute force attacks.

Key sensitivity analyzes an encryption algorithm's sensitivity to slight changes in the encryption key. A secure encryption algorithm should produce a completely different cipher image when a plaintext image is encrypted with two keys that differ only in a single bit. Furthermore, the algorithm should not be able to recover the plaintext image from a cipher image with the key changed by 1 bit. To evaluate the proposed algorithm's key sensitivity, we have encrypted the color image Sailboat ( $512 \times 512$ ) twice using two keys  $K_1$  and  $K_2$  which differ only in 1 bit. The proposed algorithm produces two completely different cipher images  $C_1$  and  $C_2$ . For the decryption process, the plaintext image is not recoverable when trying to decrypt  $C_1$  using  $K_2$ , or  $C_2$  using  $K_1$ . To quantify the key sensitivity analysis, we have calculated NPCR (Eq. 1.29) and UACI (Eq. 1.30) values for the cipher images. The NPCR value of encrypted images  $C_1$  and  $C_2$  when encrypted with 1-bit changed key is 99.6465, and the UACI value is 33.4942, which are very close to the ideal values for both metrics. Fig. 2.10 illustrates the key sensitivity test for the proposed algorithm. The results show that when encrypting the same plaintext image with keys that differ only by 1 bit, the algorithm produces completely different cipher images. Furthermore, results reveal that the plaintext image is not recoverable when using the 1-bit changed key. These results verify that the proposed algorithm has high sensitivity to minor changes in secret key.



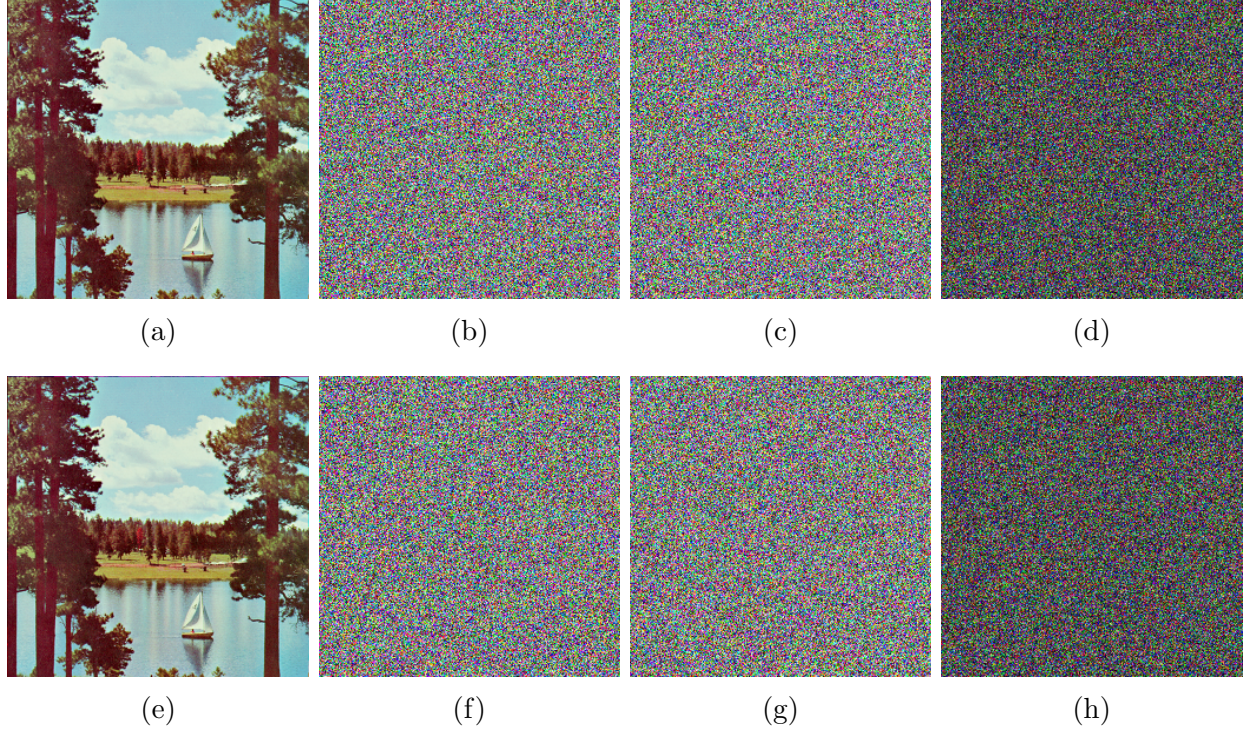


Figure 2.10: Results of key sensitivity tests on the color image Sailboat ( $512 \times 512$ ): (a) plaintext image, (b)  $C_1$  encrypted using key  $K_1$ , (c)  $C_2$  encrypted using key  $K_2$ , (d) difference of images:  $|(b)-(c)|$  (e)  $C_1$  decrypted using key  $K_1$ , (f)  $C_1$  decrypted using key  $K_2$ , (g)  $C_2$  decrypted using key  $K_1$ , (h) difference of images:  $|(f)-(g)|$ .

### 2.3.12 NIST SP 800-22 Test

We have utilized NIST SP 800-22 test suite<sup>91</sup> (version 2.1.2) to check the randomness characteristics and measure the strength property of a random sequence where the sequence should be unpredictable. Table 2.26 presents the results of the NIST SP 800-22 tests for the encrypted Baboon image, which is encrypted through our proposed algorithm. Results in Table 2.26 indicate that the sequence of the cipher image Baboon passes all NIST SP 800-22 tests.

## 2.4 Conclusion

In this chapter, an efficient and secure lightweight chaos-based image encryption algorithm is proposed. Our proposed algorithm's encryption and decryption processes use logistic maps to produce the chaotic permutation and XOR keys, and AES S-Box to substitute the val-

Table 2.26: Results of NIST SP 800-22 tests for the encrypted Baboon image.

| Test                      |         | p-Value  | Passed |
|---------------------------|---------|----------|--------|
| Frequency                 |         | 0.213309 | ✓      |
| Block Frequency           |         | 0.350485 | ✓      |
| Cumulative Sums           | Reverse | 0.350485 | ✓      |
| Cumulative Sums           | Forward | 0.534146 | ✓      |
| Runs                      |         | 0.066882 | ✓      |
| Longest Run of Ones       |         | 0.534146 | ✓      |
| Rank                      |         | 0.534146 | ✓      |
| FFT                       |         | 0.739918 | ✓      |
| Non Overlapping Template  |         | 0.911413 | ✓      |
| Overlapping Template      |         | 0.911413 | ✓      |
| Universal                 |         | 0.122325 | ✓      |
| Approximate Entropy       |         | 0.213309 | ✓      |
| Random Excursions         |         | 0.742591 | ✓      |
| Random Excursions Variant |         | 0.728588 | ✓      |
| Serial                    | Test 1  | 0.739918 | ✓      |
| Serial                    | Test 2  | 0.213309 | ✓      |
| Linear Complexity         |         | 0.350485 | ✓      |

ues of the pixels of the image to further randomize the produced cipher image. We have analyzed and compared our proposed image encryption algorithm versus other encryption methods using a variety of security, quality, and efficiency metrics. Experimental results reveal that the proposed algorithm has the least execution time as compared to other encryption methods. Results show that the proposed algorithm is up to  $15.33\times$  faster as compared to other contemporary encryption methods. Experimental results further show that the proposed algorithm produces promising results in correlation coefficient, histogram, chi-square, entropy, homogeneity, contrast, and energy analysis. Encryption quality tests, namely, mean square error, mean absolute error, peak signal-to-noise ratio, maximum deviation, irregular deviation, and deviation from the uniform histogram demonstrate that the proposed algorithm yields better quality encryption as compared to other image encryption methods. The proposed algorithm also shows high resistance against differential attack when tested for avalanche effect, NPCR, and UACI. Furthermore, the proposed algorithm provides high resistance against noise and data loss attacks. Results reveal that our proposed encryption/decryption algorithm is able to recover the plaintext image when decrypting the

cipher image with up to 50% data loss and up to 50% of pixels affected by noise. We have used NIST SP 800-22 test suite to test the randomness characteristics of the cipher image produced by the proposed algorithm, and results show that the cipher image encrypted by our proposed algorithm passes all the tests in the NIST SP 800-22 test suite.

# Chapter 3

## Image Encryption Based on Trivium

### 3.1 Introduction

Over the past few decades, image encryption algorithms have been the focus of researchers in the information security field. With the current increase in multimedia data sharing, such as images and videos, over the communication networks, a need for protecting the security and privacy of such data with real-time encryption arises. Multimedia data usually contains private and personal information, which needs to be protected from unauthorized access, and transmitted securely. While the well-established and secure encryption methods, such as Advanced Encryption Standard (AES)<sup>4</sup> and Data Encryption Standard (DES)<sup>92</sup> are recommended and widely used, they are not suitable for real-time multimedia data encryption because of their high computational requirements and slow encryption speed. Thus, specialized algorithms for multimedia data encryption are needed. Digital images usually have high redundancy due to the correlation between neighboring pixels. The high redundancy refers to the fact that adjacent pixels in an image tend to have the same values, that is, the images which contain repeating colors and pattern will have the same pixel value repeated multiple times within an image. To solve the correlation and encryption speed problems associated with digital images, researchers have proposed various encryption algorithms specifically designed for image encryption. The proposed algorithms are usually



based on chaos theory<sup>8;17;69;71;77;93</sup>, DNA encoding<sup>30;82;94–96</sup>, neural networks<sup>34;37</sup>, compressive sensing<sup>41;42;44</sup>, etc.

In this chapter, we propose a multi-round image encryption algorithm based on the Trivium cipher<sup>1</sup>. In our proposed image encryption algorithm, the Trivium cipher is used to produce pseudorandom key-streams for pixel permutation and bit-level substitution in each round. The algorithm utilizes an initial key and a key schedule to generate the round keys. The Trivium cipher utilizes the round keys and an initialization vector (IV) to generate the round key-streams. Our main contributions in this article are as follows:

- Developing a new Trivium based image encryption algorithm that has multiple encryption rounds and generates the key-streams for pixel-based row permutation, pixel-based column permutation, and bit-level substitution.
- Security analysis of the proposed image encryption algorithm.
- Evaluating the proposed image encryption algorithm with a comprehensive set of evaluation metrics, and comparing the evaluation results of the proposed algorithm with the contemporary image encryption algorithms.

The rest of the chapter is organized as follows. Section 3.2 presents preliminary information related to Trivium cipher. Section 3.3 details the proposed image encryption algorithm. Security analysis of the proposed image encryption algorithm is presented in Section 3.4. Section 3.5 discusses the experimental evaluation results. Finally, Section 3.6 concludes this chapter.

## 3.2 Preliminaries

This section provides a brief overview of nonlinear-feedback shift registers (NLFSRs) and Trivium cipher.

### 3.2.1 Nonlinear-Feedback Shift Registers (NLFSR)

NLFSRs are feedback shift registers whose input bits are a non-linear function of their previous states. Feedback shift registers consists of  $n$  binary clocked storage elements, which are called stages or bits. Each stage has a state value of 0 or 1, and a feedback function determines how the stage state value is updated. At each cycle of the feedback shift register, the register's stage values are shifted by 1-bit to the right, and the value of stage 1 is updated via the feedback function. A feedback function is a mathematical function that takes the values of two or more specific stages and performs bitwise operation(s) to produce a new bit that is then fed back to the shift register. Appropriately selected stages and bitwise operations can ensure better randomness and longer cycle of generated key-streams.

### 3.2.2 Trivium Cipher

Trivium is a synchronous stream cipher that is based on an NLFSR which generates a key-stream of up to  $2^{64}$  bits using an 80-bit secret key and an 80-bit IV<sup>1</sup>. Flowchart of the Trivium cipher is depicted in Figure 3.1. Trivium cipher has three phases: initialization phase, warm-up phase, and encryption phase. In the *initialization* phase, registers of the Trivium cipher are loaded with an 80-bit key and an 80-bit IV to the 288-bit initial state denoted as  $(S_1, \dots, S_{287}, S_{288})$  and setting all remaining bits to 0, except for  $S_{286}$ ,  $S_{287}$ , and  $S_{288}$ , which are set to 1. In the *warm-up* phase, the cipher is clocked  $4 \times 288 = 1152$  times, and no cipher output (key-stream) is generated/used. The bits produced hereafter, that is, starting with the output bit of cycle 1153, form the key-stream, and this phase can be referred to as the *encryption* phase. The warm-up phase is required for randomizing the cipher sufficiently, and to make sure that the key-stream depends on both the secret key and the IV. Algorithm 2 depicts the initialization and warm-up phase of the Trivium cipher.

In the key-stream generation or *encryption* phase, the Trivium cipher's state iteratively extracts values of 15 specific state bits which are used to update 3-bits of the state and produce 1-bit of the key-stream  $z_i$ . The Trivium cipher is then continuously clocked and at each clock cycle, one key-stream bit is generated. Algorithm 3 depicts the key-stream

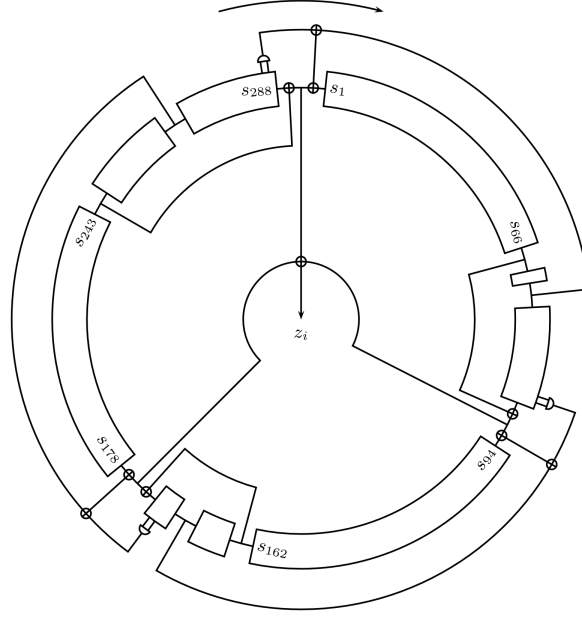


Figure 3.1: Trivium cipher flowchart<sup>1</sup>.

---

**Algorithm 2:** Trivium Initialization and Warm-up Phase<sup>1</sup>

---

```

1   $(S_1, S_1, \dots, S_{93}) \leftarrow (K_1, \dots, K_{80}, 0, \dots, 0);$ 
2   $(S_{94}, S_{95}, \dots, S_{177}) \leftarrow (IV_1, \dots, IV_{80}, 0, \dots, 0);$ 
3   $(S_{178}, S_{179}, \dots, S_{288}) \leftarrow (0, \dots, 0, 1, 1, 1);$ 
4  for  $i = 1$  to  $4 \times 288$  do
5       $t_1 \leftarrow S_{66} + S_{91} \cdot S_{92} + S_{93} + S_{171};$ 
6       $t_2 \leftarrow S_{162} + S_{175} \cdot S_{176} + S_{177} + S_{264};$ 
7       $t_3 \leftarrow S_{243} + S_{286} \cdot S_{287} + S_{288} + S_{69};$ 
8       $(S_1, S_1, \dots, S_{93}) \leftarrow (t_1, S_1, \dots, S_{92});$ 
9       $(S_{94}, S_{95}, \dots, S_{177}) \leftarrow (t_2, S_{94}, \dots, S_{176});$ 
10      $(S_{178}, S_{179}, \dots, S_{288}) \leftarrow (t_3, S_{178}, \dots, S_{287});$ 
11 end

```

---

generation process of the Trivium cipher.

### 3.3 Proposed Image Encryption Algorithm

The flowchart of the proposed Trivium-based image encryption algorithm is illustrated in Figure 3.2. The proposed algorithm utilizes a key schedule to generate the 80-bit round keys from the initial secret key  $k$ . The 80-bit round keys are then used with the 80-bit IV as the

---

**Algorithm 3:** Trivium Key-Stream Generation Phase<sup>1</sup>

---

```
1 for  $i = 1$  to  $N$  do
2    $t_1 \leftarrow S_{66} + S_{93}$ ;
3    $t_2 \leftarrow S_{162} + S_{177}$ ;
4    $t_3 \leftarrow S_{243} + S_{288}$ ;
5    $z_i \leftarrow t_1 + t_2 + t_3$ ;
6    $t_1 \leftarrow t_1 + S_{91} \cdot S_{92} + S_{171}$ ;
7    $t_2 \leftarrow t_2 + S_{175} \cdot S_{176} + S_{264}$ ;
8    $t_3 \leftarrow t_2 + S_{286} \cdot S_{287} + S_{69}$ ;
9    $(S_1, S_1, \dots, S_{93}) \leftarrow (t_1, S_1, \dots, S_{92})$ ;
10   $(S_{94}, S_{95}, \dots, S_{177}) \leftarrow (t_2, S_{94}, \dots, S_{176})$ ;
11   $(S_{178}, S_{179}, \dots, S_{288}) \leftarrow (t_3, S_{178}, \dots, S_{287})$ ;
12 end
```

---

initial parameters for the Trivium cipher to generate the rounds key-streams. Each round of the proposed algorithm consists of three steps: pixel-based row permutation, pixel-based column permutation, and bit-level substitution based on bitwise XOR operation to obscure the plaintext image values and reduce the correlation between the image pixels. The rounds key-streams are used to generate  $\text{Matrix}_i$  ( $M_i$ ) which is used for the pixel-based row and column random permutations. Each 16-bits of the binary round key-stream are converted to a decimal value, and then the **mod** of the converted decimal value is taken with the height of the image  $H$  to be suitable for use in performing the random permutations.

To illustrate the generation of the matrices  $M_i$ , let us assume that the image height  $H$  is 256, and the key-stream value is  $\{0101101000100000001010001\dots\}$  (line 3 in Algorithm 4). The first iteration of the matrix generation uses the first 16-bits of the key-stream to calculate the first value of  $M_i$  (line 5 in Algorithm 4) as  $\text{uint16}(0101101000100000) \bmod 256 = 23072 \bmod 256 = 32$ . In the second iteration, the next 16-bits of the key-stream is used to calculate the second value of  $M_i$  as  $\text{uint16}(0010100011101001) \bmod 256 = 10473 \bmod 256 = 233$ . Similarly, all the other values in  $M_i$  are calculated following the same steps (Table 3.1).

The steps of the proposed algorithm to encrypt an image are as follows:

1. Read initial secret key  $k$ .
2. Generate round keys via the key schedule using secret key  $k$  (line 2 in Algorithm 4).

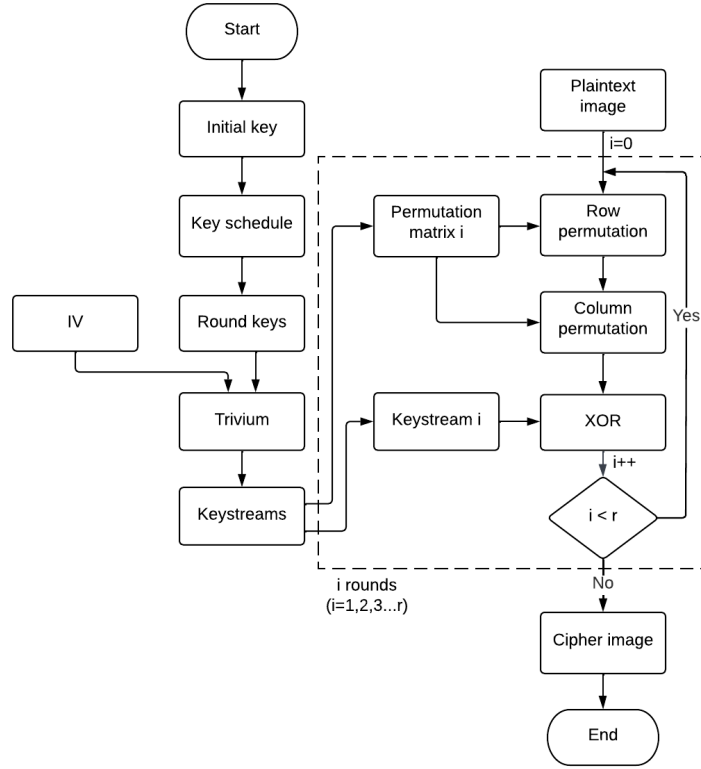


Figure 3.2: Proposed image encryption algorithm's flowchart.

Table 3.1: An example of the generation of Matrix<sub>*i*</sub> ( $M_i$ ) from the round key-stream.

|     |     |    |     |     |
|-----|-----|----|-----|-----|
| 32  | 233 | 19 | 99  | 84  |
| 176 | 107 | 98 | 82  | 38  |
| 186 | 53  | 66 | 135 | 200 |
| 117 | 209 | 6  | 142 | 122 |
| 239 | 100 | 54 | 30  | 160 |

3. Generate the IV random data block.
4. Generate key-stream<sub>*i*</sub> from Trivium using the round key<sub>*i*</sub> and the IV (line 3 in Algorithm 4).
5. Use key-stream<sub>*i*</sub> to generate permutation matrix<sub>*i*</sub> ( $M_i$ ) (lines 4-6 in Algorithm 4).
6. Read plaintext image  $P$ .
7. Use the matrix  $M_i$  to perform random row permutation on the plain image (line 9 in Algorithm 4).

---

**Algorithm 4:** Proposed Trivium-based Image Encryption

---

**Input:** Plaintext image  $P$ , initial key  $k$ , initialization vector  $IV$ , number of rounds  $r$

**Output:** Cipher image  $C$

```
1  $[H, W, nc] \leftarrow size(P)$  ; //Image height  $H$ , width  $W$ , and number of channels
    $nc$ 
2  $roundKey_{1,2,\dots,r} \leftarrow keySchedule(k)$  ;           //Generate round keys via the key
   schedule using initial key  $k$ 
3  $keyStream_{1,2,\dots,r} \leftarrow Trivium(roundKey_{1,2,\dots,r}, IV, r)$  ; //Generate key-streams
   via Trivium using  $roundKey_{1,2,\dots,r}$ ,  $IV$ ,  $r$ 
4 for  $j \leftarrow 1$  to  $H$ ,  $k \leftarrow 1$  to  $W$  do
5   |  $M_{1,2,\dots,r}(j, k) \leftarrow keyStream_{1,2,\dots,r}(j : j + 15) \bmod H$  ; //  $M_i$  denotes Matrix  $i$ ;
   |  $H$  denotes image height
6 end
7  $Q \leftarrow P$  ;           //  $Q$  denotes the current state of the image
8 for  $i \leftarrow 1$  to  $r$  do
9   |  $Q \leftarrow rowPermutation(Q, M_i)$ ;
10  |  $Q \leftarrow colPermutation(Q, M_i)$ ;
11  |  $Q \leftarrow Q \oplus keyStream_i$ ;
12 end
13  $C \leftarrow Q$ ;
return:  $C$ 
```

---

8. Use the matrix  $M_i$  to perform random column permutation on the resulting image from Step 7 (line 10 in Algorithm 4).

9. Perform bitwise XOR operation between key-stream $_i$  and the permuted image resulting from Step 8 (line 11 in Algorithm 4).

10. Repeat Steps 7, 8, and 9 until  $i = r$  ( $r$  is the total number of rounds).

11. Produce the cipher image  $C$  (line 13 in Algorithm 4).

The steps of the decryption process for the proposed algorithm are outlined below:

1. Read the cipher image  $C$  along with the initial key  $k$  and the IV.

2. Perform Steps 2 to Steps 5 of the encryption process.

3. Perform bitwise XOR operation between the key-stream $_i$  and the cipher image  $C$ .

4. Use matrix  $M_i$  to perform random column permutation on the resulting image from Step 3.
5. Use matrix  $M_i$  to perform random row permutation on the resulting image from Step 4.
6. Repeat Steps 3, 4, and 5 until  $i = r$ .
7. Produce the decrypted image  $P$ .

### 3.4 Security of The Proposed Algorithm

This section analyzes security of the proposed algorithm. The section discusses the security of each of the key aspects of the proposed algorithm, viz., Key space, the proposed key schedule and rounds, the row and column permutations, and the bitwise XOR of the image. After analyzing the security of each of the key aspects of the proposed algorithm, the overall security of the proposed algorithm is estimated.

#### 3.4.1 Key Space

An encryption algorithm should have a large key space to be able to provide sufficient security against brute-force attacks. A key space is the set of all possible keys that could be used by an encryption algorithm. Since in the proposed algorithm, round keys are generated using an initial key  $k$  of  $n = 80$  bits, the key space for the proposed algorithm is  $2^n = 2^{80}$ , which is large enough to withstand brute-force attacks. The initial key  $k$  of  $n$  bits impart a security level of  $n$  bits, where  $n = 80$  for the Trivium-based image encryption algorithm.

#### 3.4.2 Rounds and Key Schedule

The proposed algorithm has multiple rounds of encryption, where in each round a new key-stream is generated using the Trivium cipher with an 80-bit round key produced by the key schedule. By using multiple rounds  $r$ , the algorithms security increases by  $r$  times. The key schedule is designed based on a modified AES key schedule. The key schedule uses the initial

key  $k$  of length 80-bit to produce the round keys used in the Trivium cipher to generate the key-streams for each round. The number of round keys generated by the key schedule is equal to the number of rounds  $r$  in the algorithm. The key schedule leads each round to act differently and prevent slide attacks.

### 3.4.3 Pixel-Based Permutations

The row and column permutations add a level of security to the encryption by reducing the correlation between the pixels and obscuring the original image by adding randomness. The permutation is made by changing the location of the pixels based on the permutation matrix which is randomly generated by the key-stream produced by the Trivium cipher. In the proposed algorithm, there are  $(H \times W)!$  possible permutations for a given image with height  $H$  and width  $W$ <sup>97</sup>. A security level of  $m$  bits is imparted by the row and column permutations of the proposed algorithm if  $2^m = (H \times w)!$ , which implies that  $m = \log_2 [(H \times W)!]$ . Thus, the row and column permutations of the proposed algorithm impart a security level of  $m = \log_2 [(H \times W)!]$  bits.

### 3.4.4 Bitwise XOR of Image Bits

The last security step of the proposed algorithm is to take a bitwise XOR of the permuted image with the key-stream bits generated by the Trivium cipher. Bitwise XORing an image of size  $H \times W$  assuming each pixel is  $l$  bits adds a security level of  $(H \times W \times l)$  to the proposed image encryption.

### 3.4.5 Overall Security Level of the Proposed Image Encryption Algorithm

Overall security level of the proposed image encryption algorithm is estimated by summing the security level imparted by each key aspect of the proposed algorithm. Thus, the overall



security level  $\mathcal{L}$  of the proposed image encryption algorithm is given as:

$$\mathcal{L} = r \times [n + \log_2 \{(H \times W)!\} + (H \times W \times l)] \text{ bits}, \quad (3.1)$$

where  $r$  is the number of rounds,  $n = 80$  is the initial key length for Trivium cipher,  $l$  denotes the number of bits used to represent a pixel, and  $H$  and  $W$  are the height and width of the image, respectively.

## 3.5 Results and Analysis

We have evaluated the proposed algorithm using images from the USC-SIPI image dataset<sup>?</sup>. The proposed algorithm has been assessed using both 8-bit grayscale images and 24-bit color images with variable resolutions ranging from  $128 \times 128$ ,  $256 \times 256$ ,  $512 \times 512$ , to  $1024 \times 1024$  pixels. Fig. 3.3 shows a sample set of plaintext images and their corresponding cipher images.

The proposed algorithm has multiple encryption rounds and can produce cipher images using 1 round up to 15 rounds; however, all the test results in this chapter are produced using 5 rounds of encryption. The proposed algorithm is analyzed and compared to related algorithms from the literature for a variety of metrics, such as correlation coefficient, histogram analysis tests (chi-square, maximum deviation, irregular deviation, and deviation from uniform histogram), information entropy (global and local), gray-level co-occurrence matrix (GLCM) analysis (contrast, energy, and homogeneity), encryption quality analysis (mean square error (MSE), mean absolute error (MAE), and peak signal-to-noise ratio (PSNR)), resistance to differential attacks, number of pixels change rate (NPCR), unified average changing intensity (UACI), resistance to noise and data loss attacks, and the algorithm's key sensitivity. Each of these tests and their results are discussed in detail below.

### 3.5.1 Correlation Coefficient Analysis

The proposed algorithm is evaluated for the horizontal, vertical, and diagonal correlation coefficients between adjacent pixels of a cipher image (Section 3.5.1), and the horizontal,

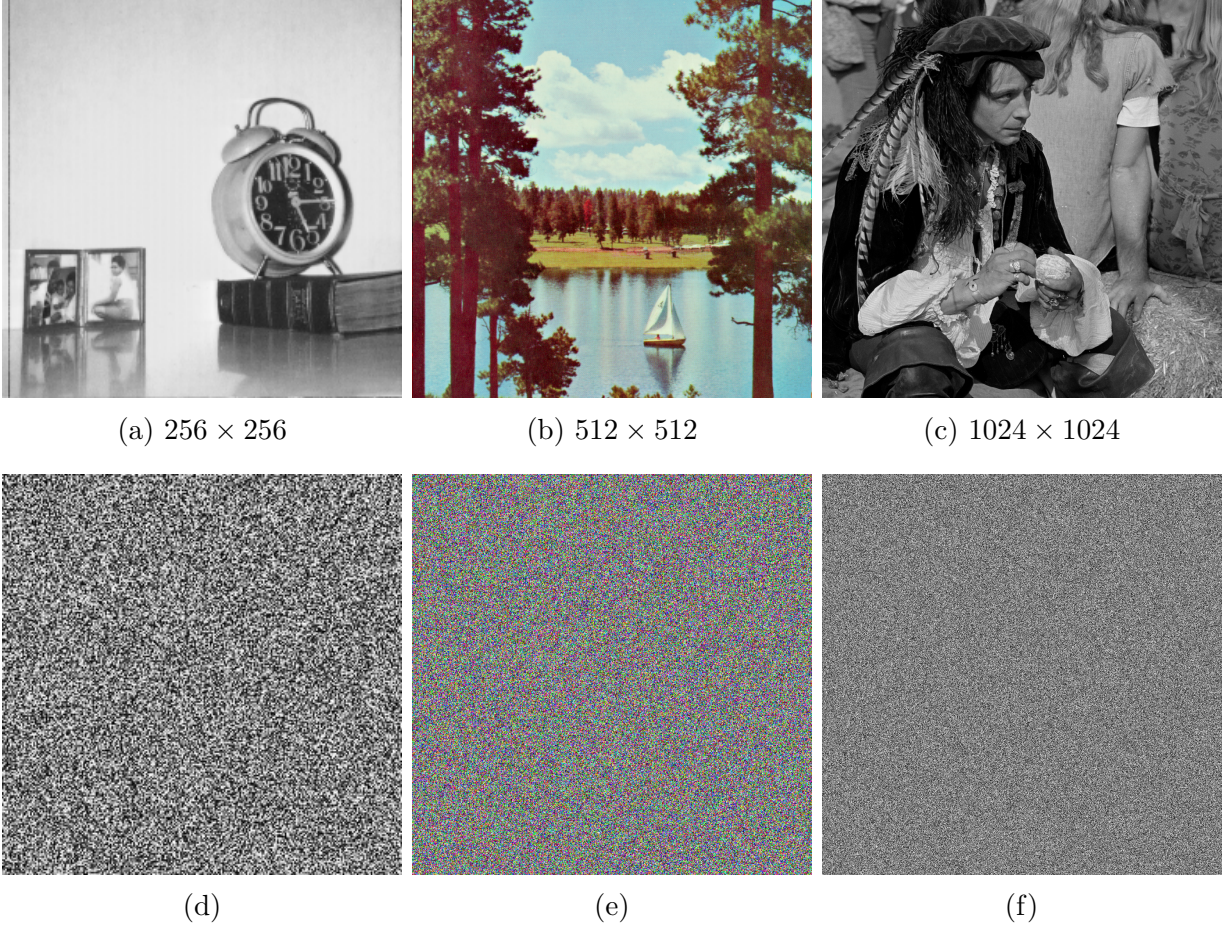


Figure 3.3: Sample images and their respective encrypted ciphers: **(a)** Clock, **(b)** Sailboat, **(c)** Male, **(d)** Clock cipher, **(e)** Sailboat cipher, **(f)** Male cipher.

vertical, and diagonal correlation coefficients between the plaintext image and its cipher (Section 3.5.1).

### Correlation Coefficient of Adjacent Pixels

Table 3.2 presents the results for the vertical, horizontal, and diagonal correlation coefficients of adjacent pixels for the cipher images. The average of the algorithm's correlation coefficient of adjacent pixels is calculated as an absolute average, that is, any deviation from zero means the presence of correlation (positive or negative) between the two compared pixels. Table 3.3 depicts a comparison of the vertical, horizontal, and diagonal correlation coefficients of the encrypted Baboon image for the proposed algorithm and other related encryption algorithms in the literature. Comparison results show that the proposed algorithm has smaller values

of correlation coefficients than the contemporary image encryption algorithms. Even though the vertical correlation coefficient of the encryption algorithm proposed in <sup>8</sup> is better than the proposed algorithm, the vertical correlation coefficient of the proposed algorithm is extremely close to 0 which means that the correlation between the adjacent vertical pixels is negligible, and thus the proposed algorithm is resilient to statistical attacks.

Table 3.2: Vertical, horizontal, and diagonal correlation coefficients of adjacent pixels of encrypted images.

| Image                  | Size and Color   | Correlation Coefficient |            |          |
|------------------------|------------------|-------------------------|------------|----------|
|                        |                  | Vertical                | Horizontal | Diagonal |
| Baboon                 | 512 × 512 Color  | -0.0005                 | 0.0004     | 0.0010   |
| Cameraman              | 256 × 256 Gray   | 0.0003                  | -0.0004    | -0.0004  |
| Clock                  | 256 × 256 Gray   | 0.0003                  | 0.0003     | -0.0020  |
| Male                   | 1024 × 1024 Gray | 0.0003                  | 0.0004     | 0.0004   |
| Peppers                | 256 × 256 Color  | 0.0020                  | 0.0005     | 0.0006   |
| Sailboat               | 512 × 512 Color  | 0.0001                  | -0.0009    | 0.0001   |
| Average <sub>abs</sub> |                  | 0.0006                  | 0.0005     | 0.0008   |

Table 3.3: Vertical, horizontal, and diagonal correlation coefficients of the encrypted Baboon image for the proposed algorithm compared to different encryption methods.

|               | Correlation Coefficient |               |               |
|---------------|-------------------------|---------------|---------------|
|               | Vertical                | Horizontal    | Diagonal      |
| Proposed      | 0.0006                  | <b>0.0005</b> | <b>0.0007</b> |
| <sup>8</sup>  | − <b>0.0001</b>         | 0.0006        | −0.0021       |
| <sup>69</sup> | −0.0086                 | 0.0023        | 0.0402        |
| <sup>71</sup> | −0.0036                 | −0.0019       | −0.0033       |
| <sup>77</sup> | −0.0004                 | −0.0007       | 0.0029        |

### Correlation Coefficient Between Plaintext and Cipher Images

The vertical, horizontal, and diagonal correlation coefficients between the plaintext images and their cipher images for the proposed algorithm are presented in Table <sup>3.4</sup>. The average of the algorithm’s correlation coefficient between a plaintext and its cipher is calculated as an absolute average, that is, any deviation from zero is means the presence of correlation

between the two compared pixels. The results show that the proposed algorithm's correlation coefficient values between the plaintext and cipher images are close to 0, which means that the cipher images have negligible to no correlation with their corresponding plaintext images.

Table 3.4: Vertical, horizontal, and diagonal correlation coefficients of plaintext and cipher images.

| Image                  | Size and Color   | Correlation Coefficient |            |          |
|------------------------|------------------|-------------------------|------------|----------|
|                        |                  | Vertical                | Horizontal | Diagonal |
| Baboon                 | 512 × 512 Color  | -0.0141                 | -0.0123    | 0.0003   |
| Cameraman              | 256 × 256 Gray   | -0.0349                 | -0.0251    | 0.0012   |
| Clock                  | 256 × 256 Gray   | -0.0607                 | 0.0119     | 0.0018   |
| Male                   | 1024 × 1024 Gray | 0.0011                  | -0.0060    | 0.0004   |
| Peppers                | 256 × 256 Color  | -0.0102                 | -0.0052    | 0.0004   |
| Sailboat               | 512 × 512 Color  | 0.0000                  | 0.0000     | 0.0003   |
| Average <sub>abs</sub> |                  | 0.0202                  | 0.0101     | 0.0007   |

### 3.5.2 Histogram Analysis

In the image encryption field, histogram analysis is used to evaluate the uniformity of the cipher image's histogram. The histogram of an image is an important index that reflects the distribution of the values of an image's pixels. A cipher image with a uniform histogram prevents attackers from extracting useful information from the cipher image. Figure 3.4 shows histograms of sample plaintext images and their respective cipher images. It can be seen from Figure 3.4 that the histograms of the cipher images are almost uniformly distributed. Hence, the proposed algorithm is effective in hiding the plaintext image's information. Thus, is considered to be secure against histogram analysis attacks. To quantify histogram analysis, a set of metrics are used, namely chi-square ( $\chi^2$ ), maximum deviation, irregular deviation, and deviation from the uniform histogram.

#### Chi-Square ( $\chi^2$ )

The  $\chi^2$  test results listed in Table 3.5 indicate that the images encrypted by the proposed algorithm have extremely low  $\chi^2$  values, and are nearly uniformly distributed. Table 3.6

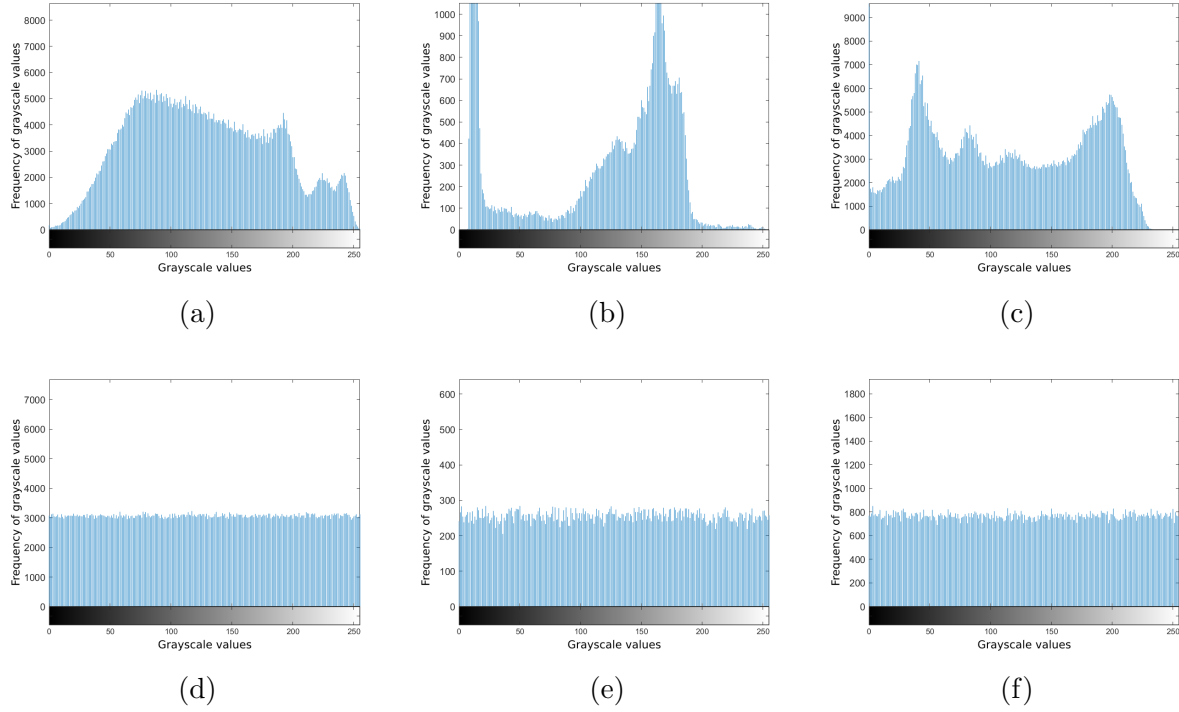


Figure 3.4: Histograms of images and their respective ciphers: (a) Baboon histogram, (b) Cameraman histogram, (c) Peppers histogram, (d) Baboon cipher histogram, (e) Cameraman cipher histogram, (f) Peppers cipher histogram.

compares  $\chi^2$  test results of the proposed algorithm to other contemporary image encryption algorithms. Results in Table 3.6 show that the proposed algorithm is better than the contemporary algorithms with respect to  $\chi^2$  metric.

### Maximum Deviation

Table 3.7 compares the maximum deviation for cipher images encrypted by the proposed algorithm with contemporary encryption algorithms. From the results, the cipher images produced by the proposed algorithm are highly deviated from the original images. Thus have a better encryption quality as compared to the cipher images produced by other encryption algorithms in the literature.

Table 3.5: Chi-Square test values for cipher images produced by our proposed algorithm.

| Image     | Size and Color   | Chi-square |
|-----------|------------------|------------|
| Baboon    | 512 × 512 Color  | 241.091    |
| Cameraman | 256 × 256 Gray   | 218.547    |
| Clock     | 256 × 256 Gray   | 213.797    |
| Male      | 1024 × 1024 Gray | 215.956    |
| Peppers   | 256 × 256 Color  | 241.974    |
| Sailboat  | 512 × 512 Color  | 239.318    |
| Average   |                  | 228.447    |

Table 3.6: Chi-Square test values of the proposed algorithm compared to different encryption algorithms.

|            | Proposed       | 8       | 96      | 98      | 99      |
|------------|----------------|---------|---------|---------|---------|
| Chi-square | <b>228.447</b> | 273.271 | 252.000 | 265.000 | 243.969 |

### Irregular Deviation

Table 3.8 presents the irregular deviation of cipher images encrypted by the proposed algorithm and compares it with related encryption algorithms. From the results in Table 3.8, it can be observed that the irregular deviation values of the cipher images produced by the proposed algorithm are small. Even though the performance of Belazi et al.<sup>85</sup> is slightly better than the proposed algorithm for some cipher images, the proposed algorithm has a better average value than all the compared algorithms.

Table 3.7: Maximum deviation results of the proposed algorithm as compared to the other algorithms.

| Image     | Proposed      | 8      | 71     | 85    |
|-----------|---------------|--------|--------|-------|
| Baboon    | <b>438546</b> | 363121 | 199158 | -     |
| Cameraman | <b>65235</b>  | 64382  | 64998  | 18007 |
| Clock     | <b>62822</b>  | -      | -      | 24823 |
| Peppers   | <b>256837</b> | 209618 | 146408 | 22935 |
| Sailboat  | <b>581918</b> | -      | -      | 25442 |
| Average   | <b>319498</b> | 167482 | 102002 | 20109 |



Table 3.8: Irregular deviation results of the proposed algorithm as compared to the other encryption algorithms.

| Image     | Proposed     | 8     | 71     | 85           |
|-----------|--------------|-------|--------|--------------|
| Baboon    | <b>59876</b> | 59921 | 80203  | -            |
| Cameraman | <b>32111</b> | 32165 | 32706  | 39224        |
| Clock     | 47060        | -     | -      | <b>27442</b> |
| Peppers   | <b>23901</b> | 76075 | 84465  | 35088        |
| Sailboat  | 72916        | -     | -      | <b>36226</b> |
| Average   | <b>39311</b> | 91460 | 129253 | 40739        |

### Deviation from Uniform Histogram

Table 3.9 presents the deviation from uniform histogram values for cipher images encrypted by the proposed algorithm and compares it with related encryption algorithms. Results in Table 3.9 show that the values of deviation from the uniform histogram of the cipher images produced by the proposed algorithm are small. Even though the performance of the encryption algorithm in <sup>8</sup> is slightly better in terms of deviation from the uniform histogram metric for some cipher images, the proposed algorithm has a better average value than all the compared algorithms.

Table 3.9: Deviation from uniform histogram of the proposed algorithm as compared to other encryption algorithms.

| Image     | Proposed      | 8             | 71     | 85     |
|-----------|---------------|---------------|--------|--------|
| Baboon    | <b>0.0242</b> | 0.0256        | 0.9990 | -      |
| Cameraman | 0.0477        | <b>0.0305</b> | 0.9961 | 0.0942 |
| Clock     | <b>0.0445</b> | -             | -      | 0.0949 |
| Peppers   | 0.0479        | <b>0.0315</b> | 0.9990 | 0.0917 |
| Sailboat  | <b>0.0241</b> | -             | -      | 0.0958 |
| Average   | <b>0.0333</b> | 0.0382        | .9984  | 0.0940 |

### 3.5.3 Entropy Analysis

Table 3.10 presents the entropy analysis of the proposed algorithm for different cipher images. Results in Table 3.10 show that the entropy values of the cipher images produced by our

proposed algorithm are extremely close to the ideal entropy. Table 3.11 presents a comparison of the global entropy of the cipher images produced by our proposed encryption algorithm to that of other encryption algorithms in literature. Results show that the global entropy of the cipher images encrypted by the encryption algorithm in<sup>8</sup> is better than our proposed encryption algorithm but the difference is negligibly small (i.e., 0.00125%).

Table 3.10: Global and local entropy values of images encrypted by the proposed algorithm.

| Image     | Size and Color   | Entropy |        |
|-----------|------------------|---------|--------|
|           |                  | Global  | Local  |
| Baboon    | 512 × 512 Color  | 7.9998  | 7.8949 |
| Cameraman | 256 × 256 Gray   | 7.9996  | 7.8948 |
| Clock     | 256 × 256 Gray   | 7.9995  | 7.9026 |
| Male      | 1024 × 1024 Gray | 7.9999  | 7.8883 |
| Peppers   | 256 × 256 Color  | 7.9992  | 7.8880 |
| Sailboat  | 512 × 512 Color  | 7.9998  | 7.8924 |
| Average   |                  | 7.9996  | 7.8935 |

Table 3.11: Global entropy of cipher image Baboon produced by the proposed algorithm compared to other encryption algorithms.

| Entropy | Proposed | 8             | 69     | 71     | 77     |
|---------|----------|---------------|--------|--------|--------|
| Global  | 7.9996   | <b>7.9997</b> | 7.9024 | 7.9992 | 7.9971 |

### 3.5.4 Gray-Level Co-Occurrence Matrix (GLCM) Analysis

The GLCM is helpful in feature extraction and texture analysis. From the GLCM, statistical measures are extracted that provide information about the image's texture features. These measures include homogeneity, contrast, and energy. In the following, we discuss these measurements for the proposed image encryption algorithm along with their comparison with other image encryption algorithms.



## Homogeneity

The homogeneity analysis results for plaintext images and their cipher images produced by our proposed encryption algorithm are listed in Table 3.12. Results in Table 3.12 indicate that the proposed algorithm produces low values of homogeneity.

Table 3.12: Homogeneity analysis for plaintext and cipher images produced by the proposed encryption algorithm.

| Image     | Size and Color   | Plaintext Image | Cipher Image |
|-----------|------------------|-----------------|--------------|
| Baboon    | 512 × 512 Color  | 0.7653          | 0.3899       |
| Cameraman | 256 × 256 Gray   | 0.8953          | 0.3889       |
| Clock     | 256 × 256 Gray   | 0.9131          | 0.3897       |
| Male      | 1024 × 1024 Gray | 0.8979          | 0.3893       |
| Peppers   | 256 × 256 Color  | 0.8837          | 0.3894       |
| Sailboat  | 512 × 512 Color  | 0.8475          | 0.3898       |
| Average   |                  |                 | 0.3895       |

## Contrast

The contrast analysis results for plaintext images and their cipher images are presented in Table 3.13. The results show that cipher images produced by the proposed algorithm have high levels of contrast.

Table 3.13: Contrast analysis for plaintext and cipher images produced by the proposed encryption algorithm.

| Image     | Size and Color   | Plaintext Image | Cipher Image |
|-----------|------------------|-----------------|--------------|
| Baboon    | 512 × 512 Color  | 0.74257         | 10.4995      |
| Cameraman | 256 × 256 Gray   | 0.58716         | 10.5647      |
| Clock     | 256 × 256 Gray   | 0.36036         | 10.5433      |
| Male      | 1024 × 1024 Gray | 0.25280         | 10.5170      |
| Peppers   | 256 × 256 Color  | 0.42068         | 10.5549      |
| Sailboat  | 512 × 512 Color  | 0.41400         | 10.5347      |
| Average   |                  |                 | 10.5357      |

## Energy

The energy analysis results for plaintext images and their cipher images produced by the proposed algorithm are presented in Table 3.14. The results show that the cipher images produced by the proposed algorithm have low values of energy.

Table 3.15 compares the average values of homogeneity, contrast, and energy of the cipher images produced by the proposed algorithm against other encryption algorithms. The results reveal that the proposed algorithm produces excellent values for these metrics when compared to other encryption algorithms; however, the energy value in<sup>89</sup> is slightly lower (i.e., by 0.128%) than the proposed algorithm. Furthermore, the homogeneity value in<sup>8</sup> is also marginally lower (i.e., by 1.49%) than the proposed algorithm.

Table 3.14: Energy analysis for plaintext and cipher images produced by the proposed algorithm.

| Image     | Size and Color   | Plaintext Image | Cipher Image |
|-----------|------------------|-----------------|--------------|
| Baboon    | 512 × 512 Color  | 0.06399         | 0.01563      |
| Cameraman | 256 × 256 Gray   | 0.18054         | 0.01564      |
| Clock     | 256 × 256 Gray   | 0.20371         | 0.01564      |
| Male      | 1024 × 1024 Gray | 0.11994         | 0.01563      |
| Peppers   | 256 × 256 Color  | 0.14053         | 0.01564      |
| Sailboat  | 512 × 512 Color  | 0.11551         | 0.01563      |
| Average   |                  |                 | 0.01563      |

Table 3.15: Comparison of homogeneity, contrast, and energy values of cipher images produced by the proposed algorithm against other encryption algorithms.

|             | Proposed       | 8             | 82      | 89             | 88      |
|-------------|----------------|---------------|---------|----------------|---------|
| Contrast    | <b>10.5357</b> | 10.5081       | 10.5079 | 8.6448         | 8.3301  |
| Energy      | 0.01563        | 0.01563       | 0.01562 | <b>0.01561</b> | 0.01760 |
| Homogeneity | 0.3895         | <b>0.3837</b> | 0.3895  | 0.4110         | 0.4208  |

### 3.5.5 Encryption Quality

The quality of the image encryption is evaluated by various metrics, such as mean square error (MSE), mean absolute error (MAE), and peak signal-to-noise ratio (PSNR). We have

evaluated our proposed image encryption algorithm for these metrics. The results of these tests for our proposed image encryption algorithm as well as comparison with contemporary encryption algorithms are detailed below.

### Mean Square Error

Table 3.16 lists the MSE values between the plaintext images and their cipher images for the proposed algorithm. Results indicate that the proposed algorithm yields high MSE values (i.e.,  $\geq 30$ ), and thus performs high quality encryption. Table 3.17 presents a comparison of the proposed image encryption algorithm to other image encryption algorithms in literature in terms of the MSE metric. The comparison results show that cipher images produced by the proposed algorithm have better encryption quality than other encryption algorithms; however, encryption algorithm in <sup>8</sup> performs slightly better (i.e., by 4.75%) than the proposed algorithm in terms of the MSE metric.

Table 3.16: MSE values of images encrypted by the proposed algorithm.

| Image     | Color and size          | MSE     |
|-----------|-------------------------|---------|
| Baboon    | $512 \times 512$ Color  | 46.0810 |
| Cameraman | $256 \times 256$ Gray   | 45.3302 |
| Clock     | $256 \times 256$ Gray   | 32.4363 |
| Male      | $1024 \times 1024$ Gray | 48.1911 |
| Peppers   | $256 \times 256$ Color  | 26.8998 |
| Sailboat  | $512 \times 512$ Color  | 43.8024 |
| Average   |                         | 40.4568 |

Table 3.17: MSE values for images encrypted by the proposed algorithm compared to related methods.

|     | Proposed | 8              | 71      | 80      | 81      |
|-----|----------|----------------|---------|---------|---------|
| MSE | 40.4568  | <b>42.3794</b> | 39.6794 | 33.4275 | 40.3295 |

## Mean Absolute Error

Table 3.18 lists the MAE values between the plaintext images and their cipher images for the proposed algorithm. Results indicate that the proposed algorithm yields high MAE values, and thus performs high quality encryption. Table 3.19 compares the MAE values of the proposed algorithm with other image encryption algorithms. The comparison results show that cipher images produced by the proposed algorithm have better encryption quality than other encryption algorithms in terms of MAE metric.

Table 3.18: MAE values of images encrypted by the proposed algorithm.

| Image     | Color and size          | MAE     |
|-----------|-------------------------|---------|
| Baboon    | $512 \times 512$ Color  | 89.8921 |
| Cameraman | $256 \times 256$ Gray   | 94.1693 |
| Clock     | $256 \times 256$ Gray   | 97.7385 |
| Male      | $1024 \times 1024$ Gray | 97.6980 |
| Peppers   | $256 \times 256$ Color  | 96.9876 |
| Sailboat  | $512 \times 512$ Color  | 96.7950 |
| Average   |                         | 95.5468 |

Table 3.19: MAE values for images encrypted by the proposed algorithm compared to related methods.

|     | Proposed       | 8     | 82    | 83    | 84 |
|-----|----------------|-------|-------|-------|----|
| MAE | <b>95.5468</b> | 91.58 | 79.57 | 78.10 | 90 |

## Peak Signal-to-Noise Ratio

Results in Table 3.20 indicate that the proposed image encryption algorithm has a good quality image encryption. Table 3.21 compares the PSNR values of the proposed algorithm to other contemporary image encryption algorithms. Results in Table 3.21 reveal that the cipher images produced by the proposed algorithm have better encryption quality than other encryption algorithms, but the algorithm in <sup>8</sup> performs marginally better (i.e., by 0.13%) than the proposed algorithm for the PSNR metric.

Table 3.20: PSNR results of images encrypted by the proposed algorithm.

| Image     | Size and color          | PSNR   |
|-----------|-------------------------|--------|
| Baboon    | $512 \times 512$ Color  | 8.7966 |
| Cameraman | $256 \times 256$ Gray   | 8.4021 |
| Clock     | $256 \times 256$ Gray   | 8.2915 |
| Male      | $1024 \times 1024$ Gray | 8.2112 |
| Peppers   | $256 \times 256$ Color  | 8.2453 |
| Sailboat  | $512 \times 512$ Color  | 8.7947 |
| Average   |                         | 8.4569 |

Table 3.21: PSNR of the images encrypted by the proposed algorithm compared to other encryption algorithms.

|      | Proposed | 8             | 71     | 82     | 53     |
|------|----------|---------------|--------|--------|--------|
| PSNR | 8.4569   | <b>8.4458</b> | 9.5424 | 9.0996 | 9.7936 |

### 3.5.6 Resistance Against Differential Attacks

Differential attacks are used by cryptanalysts to determine the relationship between the plaintext image and its cipher image, by making minor changes to the plaintext image and encrypting it using the same key. To resist such attacks, an encryption algorithm must have good diffusion characteristics, where changing a pixel in a plaintext image should produce a very different cipher image. The ability of an encryption algorithm to resist differential attacks can be determined by analyzing the algorithm's performance on avalanche effect, number of pixels change rate (NPCR), and unified average changing intensity (UACI) tests. To assess the proposed algorithm's resistance against differential attacks, the same plaintext is encrypted twice. The first cipher image is produced by encrypting the original plaintext image. The second cipher image is produced by changing a random pixel's value. For changing the pixel value in the plaintext image and to assess the resistance to differential attacks for different assigned values, we have considered and evaluated both possible extreme values (0 or 255), as well as the change in the intensity of the pixel by only 1-bit. Each of these tests and their results are discussed in detail below.

## Avalanche Effect

Figure 3.5 shows the plaintext image Baboon and its resulting cipher images. Table 3.22 lists the  $MSE_{av}$  values of the two cipher images that are produced by encrypting two plaintext images that are same but differ in only 1 pixel for the proposed algorithm. Results indicate that the proposed algorithm yields high  $MSE_{av}$  values (i.e.,  $\geq 50$ ), and thus the cipher images produced by our proposed encryption algorithm are safe against differential attacks. Table 3.23 lists the  $MSE_{av}$  between two cipher images produced by encrypting the image Baboon. The first cipher image is produced by encryption the original plaintext image. The second cipher image is produced by changing a random pixel's value in the plaintext image to: (i) 0, (ii) 255, (iii) increasing the intensity of the pixel by 1-bit, and (iv) reducing the intensity of the pixel by 1-bit.

Table 3.22: Avalanche effect  $MSE_{av}$  values of images encrypted by the proposed algorithm.

| Image     | Size and Color          | $MSE_{av}$ |
|-----------|-------------------------|------------|
| Baboon    | $512 \times 512$ Color  | 57.4265    |
| Cameraman | $256 \times 256$ Gray   | 57.6061    |
| Clock     | $256 \times 256$ Gray   | 58.1382    |
| Male      | $1024 \times 1024$ Gray | 57.3568    |
| Peppers   | $256 \times 256$ Color  | 57.6522    |
| Sailboat  | $512 \times 512$ Color  | 57.5834    |
| Average   |                         | 57.6272    |

Table 3.23: Avalanche effect  $MSE_{av}$  values of the image Baboon encrypted by the proposed algorithm.

| Random pixel change intensity  | $MSE_{av}$ |
|--------------------------------|------------|
| Pixel value set to 0           | 57.2865    |
| Pixel value set to 255         | 57.4265    |
| Pixel value increased by 1-bit | 57.2591    |
| Pixel value decreased by 1-bit | 57.3810    |
| Average                        | 57.3383    |

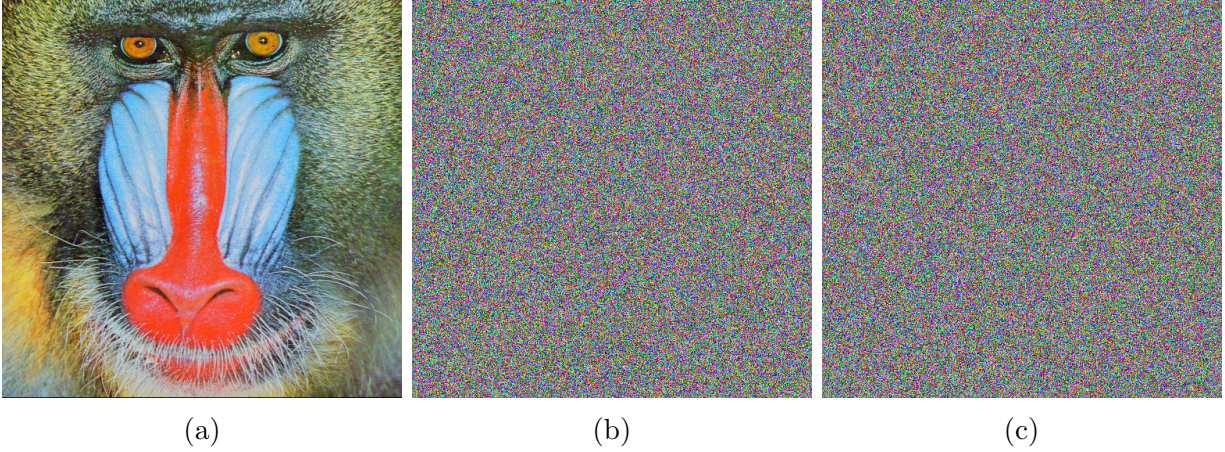


Figure 3.5: Avalanche effect on two cipher images encrypted from the same plaintext images differing in only a single pixel, (a) Baboon, (b) Baboon cipher, (c) Baboon cipher with 1 pixel changed in the Baboon image.

### Number of Pixels Change Rate

al.<sup>57</sup>. Table 3.24 lists the NPCR values for the proposed algorithm. Results indicate that the proposed algorithm yields high NPCR values, and thus the cipher images produced by the proposed encryption algorithm are safe against differential attacks. Table 3.25 lists the NPCR of the two cipher images produced by encrypting the image Cameraman. The first cipher image is produced by encryption the original plaintext image. The second cipher image is produced by changing a random pixel's value in the plaintext image to: (i) 0, (ii) 255, (iii) increasing the intensity of the pixel by 1-bit, and (iv) reducing the intensity of the pixel by 1-bit. Table 3.26 compares the NPCR values of the proposed algorithm with other image encryption algorithms. Results in Table 3.26 indicate that the NPCR values of the proposed algorithm are higher than other encryption algorithms.

### Unified Average Changing Intensity (UACI)

Table 3.27 lists the UACI values for the proposed algorithm. Results indicate that the proposed algorithm yields high UACI values, and thus is resilient against differential attacks. Table 3.28 lists the UACI of two cipher images produced by encrypting the image Clock. The first cipher image is produced by encryption the original plaintext image. The second



Table 3.24: NPCR values of images encrypted by the proposed algorithm.

| Image     | Size and Color   | NPCR    |
|-----------|------------------|---------|
| Baboon    | 512 × 512 Color  | 99.6222 |
| Cameraman | 256 × 256 Gray   | 99.6109 |
| Clock     | 256 × 256 Gray   | 99.6292 |
| Male      | 1024 × 1024 Gray | 99.6156 |
| Peppers   | 256 × 256 Color  | 99.6206 |
| Sailboat  | 512 × 512 Color  | 99.6199 |
| Average   |                  | 99.6197 |

Table 3.25: NPCR values of the image Cameraman encrypted by the proposed algorithm.

| Random pixel change intensity  | NPCR    |
|--------------------------------|---------|
| Pixel value set to 0           | 99.5727 |
| Pixel value set to 255         | 99.6109 |
| Pixel value increased by 1-bit | 99.5666 |
| Pixel value decreased by 1-bit | 99.5819 |
| Average                        | 99.5830 |

cipher image is produced by changing a random pixel's value in the plaintext image to: (i) 0, (ii) 255, (iii) increasing the intensity of the pixel by 1-bit, and (iv) reducing the intensity of the pixel by 1-bit. Table 3.29 compares the UACI values of the proposed algorithm with other image encryption algorithms. Results in Table 3.29 indicate that the UACI values of the proposed algorithm are better than other encryption algorithms.

### 3.5.7 Resistance to Noise and Data Loss

To assess the resilience of the proposed encryption algorithm against noise, we have added salt and pepper noise to the cipher image with densities of 0.15, 0.25, and 0.5. It is to be noted that a noise density of 0.1 affects approximately 10% of pixels, and noise density of

Table 3.26: NPCR of the proposed algorithm compared with other encryption methods.

|      | Proposed       | 8       | 69      | 71      | 77      |
|------|----------------|---------|---------|---------|---------|
| NPCR | <b>99.6197</b> | 99.6153 | 99.5893 | 99.6059 | 99.5743 |



Table 3.27: UACI values of images encrypted by the proposed algorithm.

| Image     | Size and Color          | UACI    |
|-----------|-------------------------|---------|
| Baboon    | 512 $\times$ 512 Color  | 33.4901 |
| Cameraman | 256 $\times$ 256 Gray   | 33.5461 |
| Clock     | 256 $\times$ 256 Gray   | 33.7048 |
| Male      | 1024 $\times$ 1024 Gray | 33.4707 |
| Peppers   | 256 $\times$ 256 Color  | 33.5548 |
| Sailboat  | 512 $\times$ 512 Color  | 33.5361 |
| Average   |                         | 33.5504 |

Table 3.28: UACI values of the image Clock encrypted by the proposed algorithm.

| Random pixel change intensity  | UACI    |
|--------------------------------|---------|
| Pixel value set to 0           | 33.4870 |
| Pixel value set to 255         | 33.7048 |
| Pixel value increased by 1-bit | 33.4079 |
| Pixel value decreased by 1-bit | 33.5179 |
| Average                        | 33.4198 |

1 affects approximately 100% of pixels. Figure 3.6 shows cipher images to which salt and pepper noise has been introduced with different densities. Results in Figure 3.6 reveal that the proposed algorithm is able to decrypt cipher images affected by salt and pepper noise with density up to 0.5. To evaluate the the robustness of the proposed algorithm against occlusion attacks, we have cropped out segments of the cipher image with different severities, that is, 10%, 25%, and 50% cropping. Figure 3.7 depicts data loss (occlusion) attacks on a cipher image with different severities. Results in Figure 3.7 reveal that the proposed algorithm is able to successfully decrypt cipher images with cropping up to 50%. It is important to note that the quality of the decrypted images obtained from the encrypted images subjected to noise and/or data loss is expected to be degraded. We have employed these attacks to

Table 3.29: UACI of the proposed algorithm compared with other encryption methods.

|      | Proposed       | 8       | 69      | 71      | 77      |
|------|----------------|---------|---------|---------|---------|
| UACI | <b>33.5504</b> | 33.4718 | 33.3730 | 33.4375 | 33.3941 |

test the algorithm's ability to decrypt a cipher image affected by noise or cropping during transmission, whether due to noisy channels or adversarial attacks. The tests confirm that the proposed algorithm can withstand such attacks and decrypt the affected cipher image.

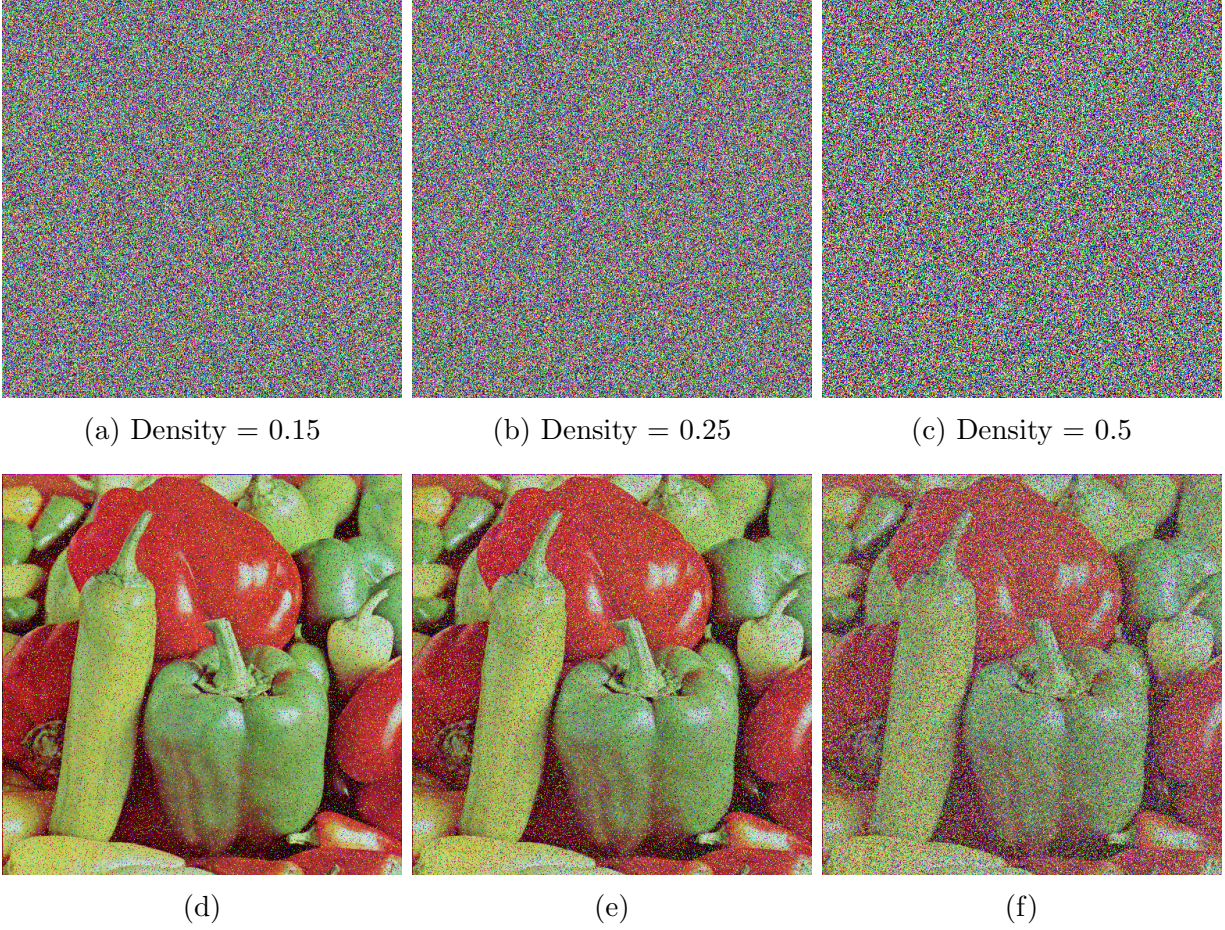


Figure 3.6: Results of salt and pepper noise attack on the color image Peppers ( $512 \times 512$ ): (a–c) different noise densities introduced, (d–f) corresponding decrypted images.

### 3.5.8 Key Sensitivity

The sensitivity of an encryption algorithm to minor changes in the secret key is a principal factor in evaluating an encryption algorithm's security. A key-sensitive encryption algorithm should not be able to decrypt a cipher image and recover the plaintext image with a key that is changed by as little as a single bit. Moreover, a change of 1-bit in the key should cause the algorithm to produce two totally different cipher images. Figure 3.8 depicts the results



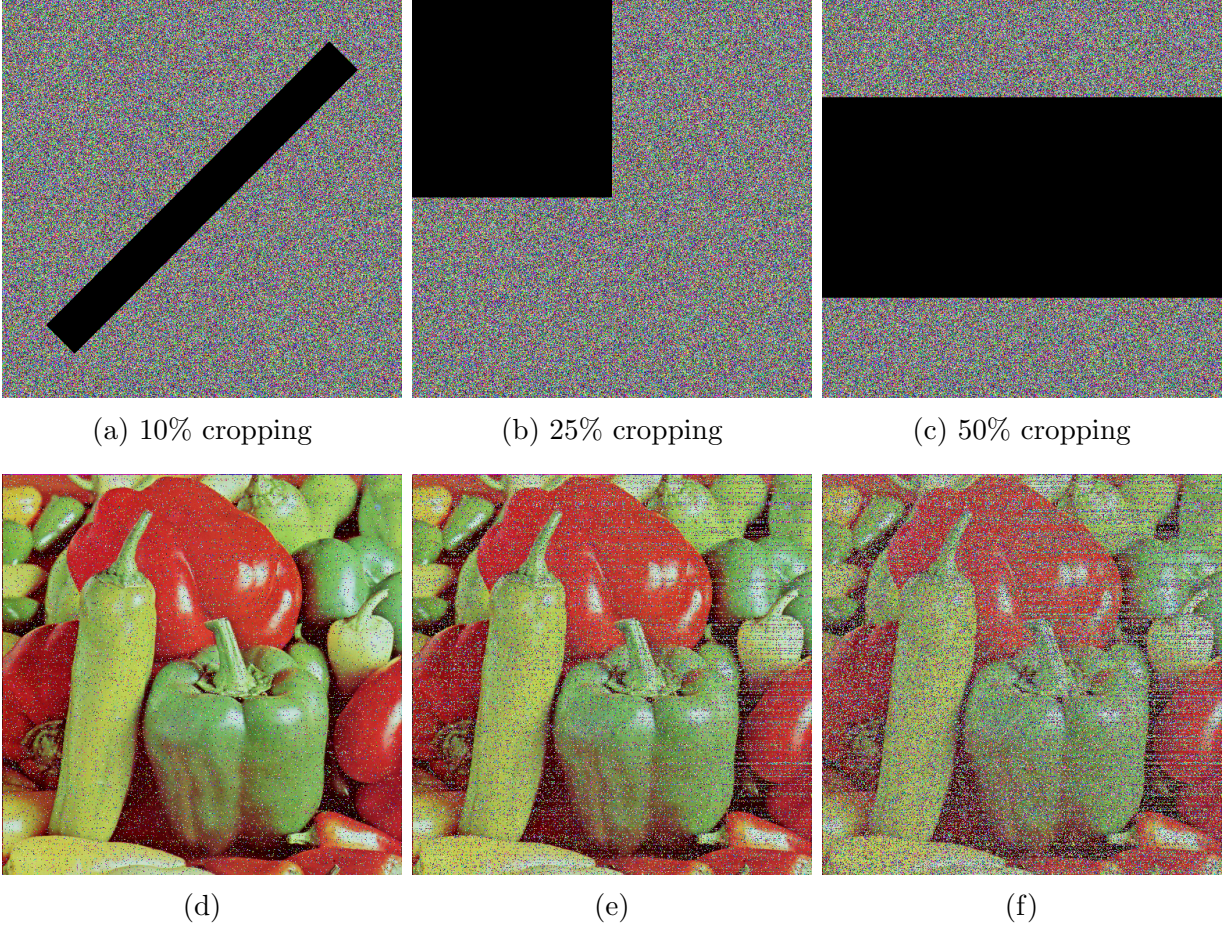


Figure 3.7: Results of occlusion attacks (data loss) on the color image Peppers ( $512 \times 512$ ): (a–c) different severities of occlusion attacks, (d–f) corresponding decrypted images.

of key sensitivity tests for the proposed algorithm. From the test results in Figure 3.8, it can be seen that the proposed algorithm is highly sensitive to changes in the secret key.

*Key sensitivity for encryption:* To evaluate the sensitivity of the proposed algorithm to slight changes in the key for encryption, we have encrypted the image Clock using two keys  $k_1$  and  $k_2$  that differ in only 1-bit. Results in Figure 3.8 (b), (c), and (d) show that the proposed algorithm produces two totally different cipher images for the keys  $k_1$  and  $k_2$ .

*Key sensitivity for decryption:* To assess the key sensitivity to changes in the key for decryption, we have encrypted the image Clock with a key  $k_1$  to produce a cipher image  $C_1$ . Results in Figure 3.8 (f), (g), and (h) show that decrypting  $C_1$  with a key  $k_2$  that differs in only 1-bit from key  $k_1$  does not recover the plaintext image.

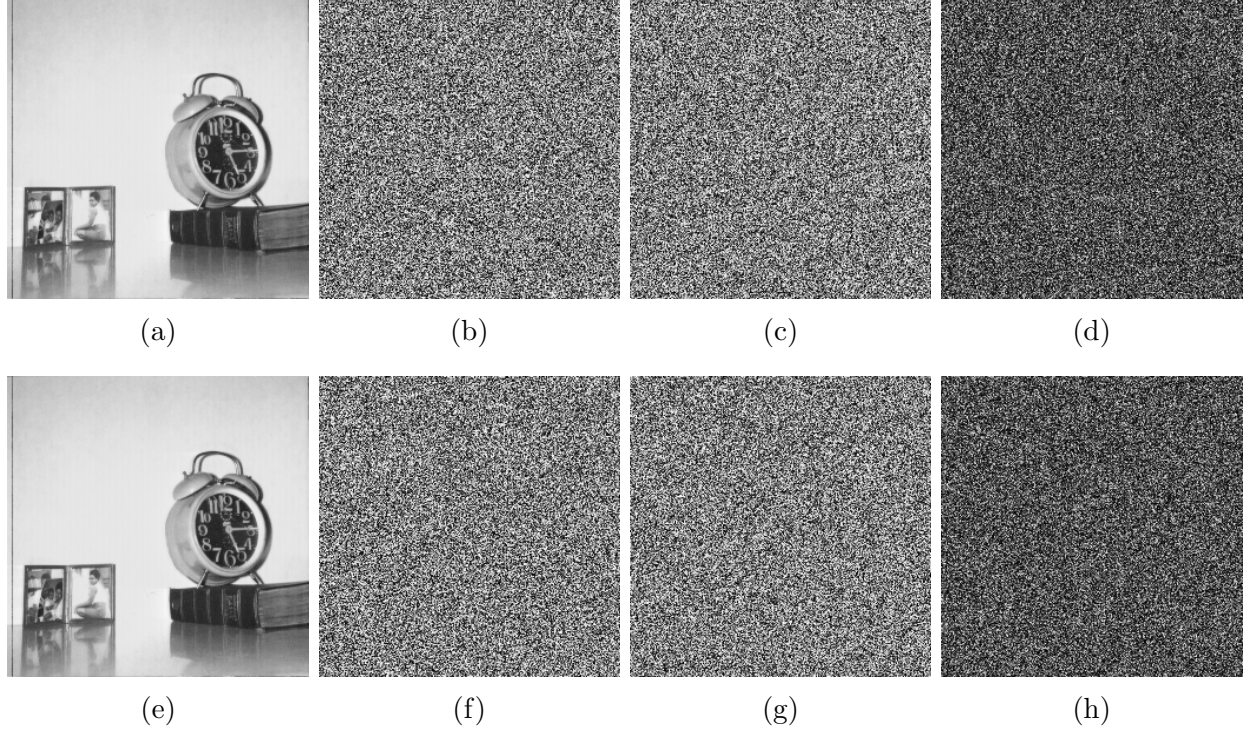


Figure 3.8: Results of key sensitivity tests on the color image Sailboat ( $512 \times 512$ ): (a) plaintext image, (b)  $C_1$  encrypted using key  $k_1$ , (c)  $C_2$  encrypted using key  $k_2$ , (d) difference of images:  $-(b) - (c)-$ , (e)  $C_1$  decrypted using key  $k_1$ , (f)  $C_1$  decrypted using key  $k_2$ , (g)  $C_2$  decrypted using key  $k_1$ , (h) difference of images:  $-(f) - (g)-$ .

### 3.5.9 NIST SP 800-22 Test

We have utilized NIST SP 800-22 test suite<sup>91</sup> (version 2.1.2) to check the randomness characteristics and measure the strength property of a random sequence where the sequence should be unpredictable. Table 3.30 presents the results of the NIST SP 800-22 tests for the encrypted Peppers image, which is encrypted through our proposed algorithm. Results in Table 3.30 indicate that the sequence of the cipher image Baboon passes all NIST SP 800-22 tests.

## 3.6 Conclusions

In this chapter, a secure multi-round image encryption algorithm based on nonlinear-feedback shift registers is proposed. The proposed algorithm uses Trivium cipher to produce key-

Table 3.30: Results of NIST SP 800-22 tests for the encrypted Baboon image.

| Test                      |         | p-Value  | Passed |
|---------------------------|---------|----------|--------|
| Frequency                 |         | 0.080519 | ✓      |
| Block Frequency           |         | 0.401199 | ✓      |
| Cumulative Sums           | Reverse | 0.275709 | ✓      |
| Cumulative Sums           | Forward | 0.867692 | ✓      |
| Runs                      |         | 0.514124 | ✓      |
| Longest Run of Ones       |         | 0.224821 | ✓      |
| Rank                      |         | 0.010988 | ✓      |
| FFT                       |         | 0.334538 | ✓      |
| Non Overlapping Template  |         | 0.249284 | ✓      |
| Overlapping Template      |         | 0.798139 | ✓      |
| Universal                 |         | 0.366918 | ✓      |
| Approximate Entropy       |         | 0.071177 | ✓      |
| Random Excursions         |         | 0.719747 | ✓      |
| Random Excursions Variant |         | 0.834308 | ✓      |
| Serial                    | Test 1  | 0.042808 | ✓      |
| Serial                    | Test 2  | 0.062821 | ✓      |
| Linear Complexity         |         | 0.678686 | ✓      |

streams from an 80-bit round key and an 80-bit IV. Each key-stream is used once for each round and is used to perform pixel-based row permutation, pixel-based column permutation, and bit-level substitution using bitwise XOR on the plaintext image to produce the cipher image. The proposed algorithm is analyzed and compared with various encryption algorithms in literature utilizing a comprehensive set of quality, efficiency, and security metrics. The experimental results reveal that the proposed algorithm produces better results in terms of correlation coefficient, chi-square, maximum deviation, irregular deviation, and deviation from the uniform histogram when compared to other contemporary image encryption algorithms. Furthermore, the image texture analysis using the GLCM's homogeneity, contrast, and energy show that the proposed algorithm yields better encryption as compared to other image encryption algorithms. When tested for encryption quality, that is, via mean square error, mean absolute error, and peak signal-to-noise ratio metrics, the proposed algorithm demonstrates excellent quality results that are similar to other contemporary image encryption methods. Additionally, experimental results reveal that the proposed algorithm is resilient against differential attacks and noise and occlusion attacks.

One of the limitations for the proposed algorithm that needs to be tested is the algorithm's resilience against quantum computing attacks. Even though the proposed algorithm increases the Trivium key space of  $2^{80}$  thus making it more difficult for attackers to break the algorithm using brute-force attacks, its security against attacks by quantum computers needs to be analysed. Moreover, the proposed algorithm needs further optimization to be compression friendly. At its current state, the algorithm cannot correctly decrypt a cipher image that was compressed after it was encrypted. The image will be decrypted but a large amount of data will be lost. Furthermore, the proposed algorithm uses a software implementation of the Trivium cipher, but the Trivium cipher delivers much faster encryption speed in hardware as compared to software. In our future work, we plan to implement the proposed algorithm in hardware using a field-programmable gate array (FPGA). Furthermore, we plan to explore acceleration or parallelization of the proposed image encryption algorithm.



# Chapter 4

## Evaluating The Ascon128 for Image Encryption

### 4.1 Introduction

Researchers in the field of information security have directed their attention towards image encryption algorithms for the last few decades. Given the current increase in sharing of multimedia data like images and videos across networks, there is a growing demand for real-time encryption algorithms to ensure the security and privacy of such data. Although traditional methods like the Advanced Encryption Standard (AES)<sup>4</sup>, RSA, etc., are well-established, secure, and highly recommended, they are unsuitable for real-time image encryption due to their substantial computational demands and slow encryption speed. Therefore, researchers have proposed specialized algorithms designed specifically for image encryption. The proposed image encryption algorithms aims to solve the high computational demands, slow encryption speed, and ensure the security and integrity of the multimedia data. The proposed algorithms are usually based on chaotic system, DNA encoding, neural networks, compressive sensing, frequency domain, and meaningful sensing.

In this chapter, we evaluate the Ascon encryption algorithm<sup>2</sup> for image encryption. In our tests, the Ascon encryption algorithm is used in authenticated encryption mode to produce

the cipher images. Our main contributions in this article are as follows:

- Testing the Ascon cipher ability to encrypt images.
- Evaluating the Ascon image encryption algorithm using a comprehensive set of assessment metrics.
- Comparison of the evaluation results of the Ascon algorithm with other image encryption algorithms.

The rest of the chapter is organized as follows. Section 4.2 presents a brief overview of the Ascon cipher. Section 4.3 discusses the experimental evaluation results. Finally, Section 4.4 concludes this chapter.

## 4.2 Ascon Cipher

Ascon is a lightweight authenticated encryption algorithm designed for secure communications in constrained environments. Ascon provides authenticated encryption with associated data (AEAD) and hashing functionality. The algorithm was the primary choice at the CAESAR competition (Competition for Authenticated Encryption: Security, Applicability, and Robustness), and was selected by National Institute of Standards and Technology (NIST) to standardize lightweight cryptography applications. Ascon was specifically developed to be efficient in both memory usage and processing power. Furthermore, it was designed for high security and robustness while having a simple and transparent design in both hardware and software implementation.

### 4.2.1 Ascon’s Authenticated Encryption Mode

Ascon uses a duplex-sponge-based mode of operation for authenticated encryption, and the sponge operates on 320-bit state  $S$ . The recommended parameters for Ascon authenticated encryption mode are listed in Table 4.1. The recommended key  $K$ , nonce  $N$ , and tag  $T$  length is 128 bits. The rate  $r$  can be either 64 or 128-bits, and the internal permutations



rounds number  $p^a$  is 12 rounds and internal permutations rounds number  $p^b$  can be either 6 or 8 rounds based on  $r$ . Ascon's encryption has four phases: initialization, associated data processing, plaintext processing, and finalization. The initialization and finalization phases use  $p^a$  number of permutations rounds. The associated data and plaintext processing phases uses  $p^b$  number of permutations rounds. Figure 4.1 illustrates the Ascon encryption operations. Each phase is discussed further in the following:

Table 4.1: Recommended parameters for Ascon authenticated encryption<sup>2</sup>.

|           | Key $K$ | Nonce $N$ | Tag $T$ | Rate $r$ | Capacity $c$ | Rounds |       |
|-----------|---------|-----------|---------|----------|--------------|--------|-------|
|           |         |           |         |          |              | $p^a$  | $p^b$ |
| Ascon128  | 128     | 128       | 128     | 64       | 256          | 12     | 6     |
| Ascon128a | 128     | 128       | 128     | 128      | 192          | 12     | 8     |

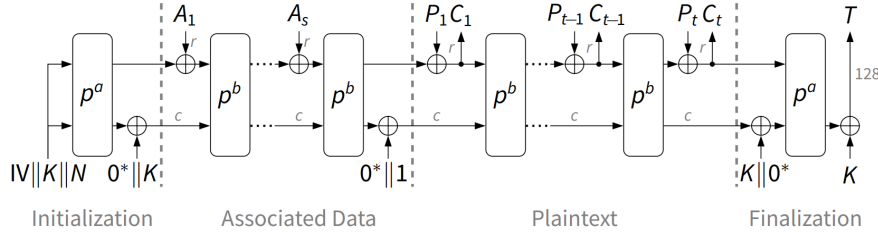


Figure 4.1: The mode of operation for Ascon authenticated encryption<sup>2</sup>.

**Initialization:** Ascon has a 320-bit initial state  $S$  which is initialized with the key  $K$ , nonce  $N$ , and an initialization vector  $IV$ . The  $IV$  is a constant that specifies the exact algorithm being used (i.e., Ascon128 or Ascon128a). The  $IV$  size is 64-bits, and consists of the key size  $k$ , rate  $r$ , and internal round numbers  $a$  and  $b$ .

$$IV_{k,r,a,b} \leftarrow k || r || a || b || 0^{160-k}$$

$$IV_{k,r,a,b} = \begin{cases} 80400c0600000000 & \text{for Ascon128} \\ 80800c0800000000 & \text{for Ascon128a} \end{cases}$$

$$S \leftarrow IV_{k,r,a,b} || K || N$$

Next,  $a$ -rounds of permutation  $p^a$  are applied to the initial state  $S$ . Then, the state  $S$  is

XORed with the secret key  $K$ .

$$S \leftarrow p^a(S) \oplus (0^{320-k} || K)$$

**Associated Data Processing:** The associated data  $A$  are processed in blocks of  $r$  bits. Each block  $A_i$  is XORed with  $S_r$  which is the first  $r$  bits of the state  $S$ . Then the  $b$ -round permutation  $p^b$  is applied to  $S$ .

$$S \leftarrow p^b((S_r \oplus A_i) || S_c), \quad 1 \leq i \leq s$$

where  $S_c$  is the capacity  $c$  part of the state  $S$ . After processing the associated data  $A$ , 1-bit domain separation constant is XORed to  $S$ .

$$S \leftarrow S \oplus (0^{319} || 1)$$

**Plaintext/Ciphertext Processing:** Plaintext  $P$  is split into  $t$  blocks  $P_i$  with  $r$ -bits size. Ascon works as a single pass over the plaintext  $P$  for  $t$  iterations on each  $P_i$  block. For encryption, in each iteration a plaintext block  $P_i$  is XORed with  $S_r$  which is the first  $r$  bits of the state  $S$  to produce ciphertext block  $C_i$ . Next, the state  $S$  is transformed by the internal permutations  $p^b$  for  $b$  rounds except for the last block. This process is repeated till all  $t$  blocks of plaintext  $P$  are processed.

$$C_i \leftarrow S_r \oplus P_i$$

$$S \leftarrow \begin{cases} p_b(C_i || S_c) & \text{if } 1 \leq i < t \\ C_i || S_c & \text{if } 1 \leq i = t \end{cases}$$

For decryption, in each iteration a ciphertext block  $C_i$  is XORed with  $S_r$  to produce plaintext block  $P_i$ . Then  $S_r$  is replaced by  $C_i$ . Next, the state  $S$  is transformed by the internal

permutations  $p^b$  for  $b$  rounds except for the last block.

$$\begin{aligned} P_i &\leftarrow S_r \oplus C_i \\ S &\leftarrow p_b(C_i || S_c), \end{aligned} \quad 1 \leq i < t$$

**Finalization:** The secret key  $K$  is XORed with state  $S$ . Then  $S$  is transformed by the permutation  $p^a$  for  $a$  rounds. Finally, a 128-bit tag  $T$  is extracted for authentication. The tag is acquired by XORing the last 128 bits of the state  $S$  with the last 128 bits of the key  $K$ . The ciphertext is decrypted only when the decryption algorithm calculated tag matches the received tag.

$$\begin{aligned} S &\leftarrow p^a(S \oplus (0^r || K || 0^{c-k})) \\ T &\leftarrow [S]^{128} \oplus [K]^{128} \end{aligned}$$

## 4.2.2 Ascon's Permutation

The Ascon permutations are substitution-permutation network (SPN)-based round transformations that consist of three steps, viz., addition of constants, nonlinear substitution layer, and linear diffusion layer. These steps operate on 320-bit state  $S$  that is divided into five 64-bit registers words  $x_i$ ,  $i \in \{0, 1, 2, 3, 4\}$ , that is,  $S = x_0 || x_1 || x_2 || x_3 || x_4$ . The number of internal permutations rounds are tunable security parameters (discussed in section 4.2.1).

**Addition of constants  $p_c$ :** In this step, a round-specific 1-byte constant is XORed with the register word  $x_2$ . Table 4.2 lists the round-specific constants.

**Nonlinear substitution layer  $p_s$ :** In this step, the state  $S$  is updated with 64 parallel applications of the 5-bit S-box ( $Sx$ ). The substitutions are applied in a bit-sliced fashion vertically across words. Table 4.3 shows the Ascon's S-box  $S$  lookup table.

**Linear diffusion layer  $p_l$ :** In this step, each 64-bit register word  $x_i$  is XORed with different horizontally rotated copies within each word. Using linear function  $\Sigma_i(x_i)$ , linear diffusion

Table 4.2: The round-specific constants for each round of  $p^a$  and  $p^b$ <sup>2</sup>.

| $p^{12}$ | $p^8$ | $p^6$ | Constant             |
|----------|-------|-------|----------------------|
| 0        |       |       | 000000000000000000f0 |
| 1        |       |       | 000000000000000000e1 |
| 2        |       |       | 000000000000000000d2 |
| 3        |       |       | 000000000000000000c3 |
| 4        | 0     |       | 000000000000000000b4 |
| 5        | 1     |       | 000000000000000000a5 |
| 6        | 2     | 0     | 00000000000000000096 |
| 7        | 3     | 1     | 00000000000000000087 |
| 8        | 4     | 2     | 00000000000000000078 |
| 9        | 5     | 3     | 00000000000000000069 |
| 10       | 6     | 4     | 0000000000000000005a |
| 11       | 7     | 5     | 0000000000000000004b |

Table 4.3: Ascon’s S-box lookup table<sup>2</sup>.

|        |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|--------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| $x$    | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | a  | b  | c  | d  | e  | f  |
| $S(x)$ | 4  | b  | 1f | 14 | 1a | 15 | 9  | 2  | 1b | 5  | 8  | 12 | 1d | 3  | 6  | 1c |
| $x$    | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 1a | 1b | 1c | 1d | 1e | 1f |
| $S(x)$ | 1e | 13 | 7  | e  | 0  | d  | 11 | 18 | 10 | c  | 1  | 19 | 16 | a  | f  | 17 |

step can be defined as follows:

$$x_0 \leftarrow \Sigma_0(x_0) = x_0 \oplus (x_0 \ggg 19) \oplus (x_0 \ggg 28)$$

$$x_1 \leftarrow \Sigma_1(x_1) = x_1 \oplus (x_1 \ggg 61) \oplus (x_1 \ggg 39)$$

$$x_2 \leftarrow \Sigma_2(x_2) = x_2 \oplus (x_2 \ggg 1) \oplus (x_2 \ggg 6)$$

$$x_3 \leftarrow \Sigma_3(x_3) = x_3 \oplus (x_3 \ggg 10) \oplus (x_3 \ggg 17)$$

$$x_4 \leftarrow \Sigma_4(x_4) = x_4 \oplus (x_4 \ggg 7) \oplus (x_4 \ggg 41)$$

### 4.3 Results and Analysis

To evaluate the Ascon algorithm, the USC-SIPI standard test images dataset is used<sup>75</sup>, namely Baboon, Cameraman, Clock, Male, Peppers, and Sailboat, as shown in Figure 4.2.

We have assessed the Ascon algorithm using both 8-bit grayscale images and 24-bit color images with variable resolutions ranging from  $128 \times 128$ ,  $256 \times 256$ ,  $512 \times 512$ , to  $1024 \times 1024$  pixels. Figure 4.3 shows a sample set of plaintext images and their respective cipher images.

We have used the Ascon algorithm in authenticated encryption mode Ascon128 using the recommended parameters ( Table 4.1). We have tested the algorithm on a machine using an Intel Core i5 CPU running at 3.5 GHz. The machine has 12 GB of RAM and runs the Windows 10 operating system. We have run the Ascon algorithm code in C language to produce the cipher and decrypted images, and have used MATLAB R2022b to evaluate the results.

We have evaluated, analyzed, and compared the Ascon algorithm to relevant algorithms from the literature using a comprehensive set of metrics, such as correlation coefficient, histogram analysis tests (chi-square, maximum deviation, irregular deviation, and deviation from uniform histogram), information entropy (global and local), gray-level co-occurrence matrix (GLCM) analysis (contrast, energy, and homogeneity), encryption quality analysis (mean square error (MSE), mean absolute error (MAE), and peak signal-to-noise ratio (PSNR)), resistance to differential attacks, number of pixels change rate (NPCR), unified average changing intensity (UACI), resistance to noise and data loss attacks, and the algorithm's key sensitivity. Furthermore, The NIST SP 800-22 test suite is also used to test the randomness characteristics of the produced cipher images. Each of these tests and their results are discussed in detail below.

### 4.3.1 Correlation Coefficient Analysis

The Ascon image encryption algorithm is evaluated for the horizontal, vertical, and diagonal correlation coefficients between adjacent pixels of a cipher image (Section 4.3.1), and the horizontal, vertical, and diagonal correlation coefficients between a plaintext image and its cipher (Section 4.3.1).

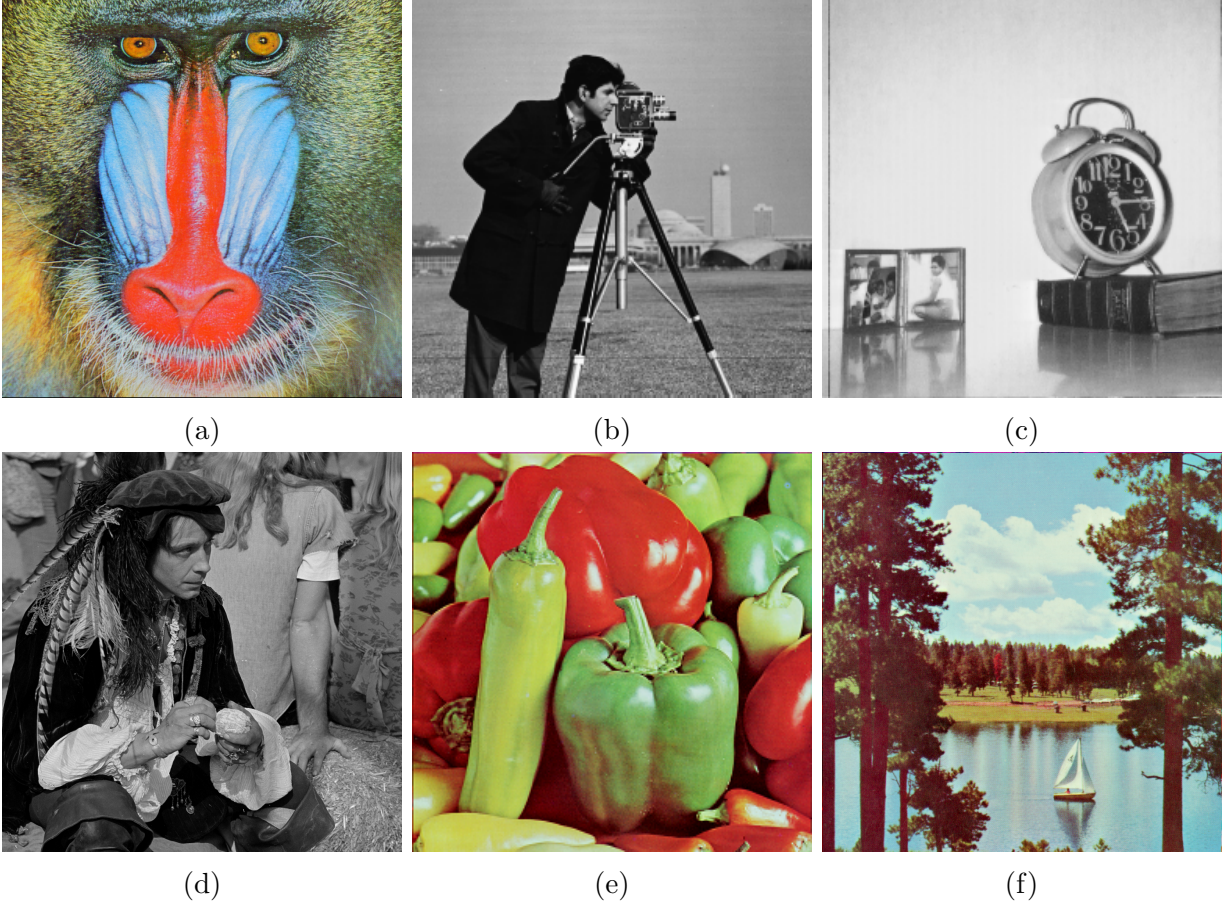


Figure 4.2: Images used to test the Ascon algorithm (a) Baboon, (b) Cameraman, (c) Clock, (d) Male, (e) Peppers, and (f) Sailboat.

### Correlation Coefficient of Adjacent Pixels

The results for the vertical, horizontal, and diagonal correlation coefficients of adjacent pixels for the cipher images are displayed in Table 4.4. Table 4.5 lists a comparison of the correlation coefficients in the vertical, horizontal, and diagonal directions between the Ascon algorithm and relevant encryption algorithms found in the literature. From the results, it can be observed that the Ascon algorithm has smaller values of correlation coefficients as compared to the other algorithms. However, the vertical correlation coefficient of the encryption algorithm by Alghamdi et al.<sup>8</sup>, and the diagonal correlation coefficient of the encryption algorithm by Alghamdi and Munir<sup>100</sup> are better than the Ascon algorithm for this metric. The average results (Table 4.4) of the correlation coefficients for the Ascon algorithm are extremely close to 0 which means that the Ascon algorithm greatly reduces the



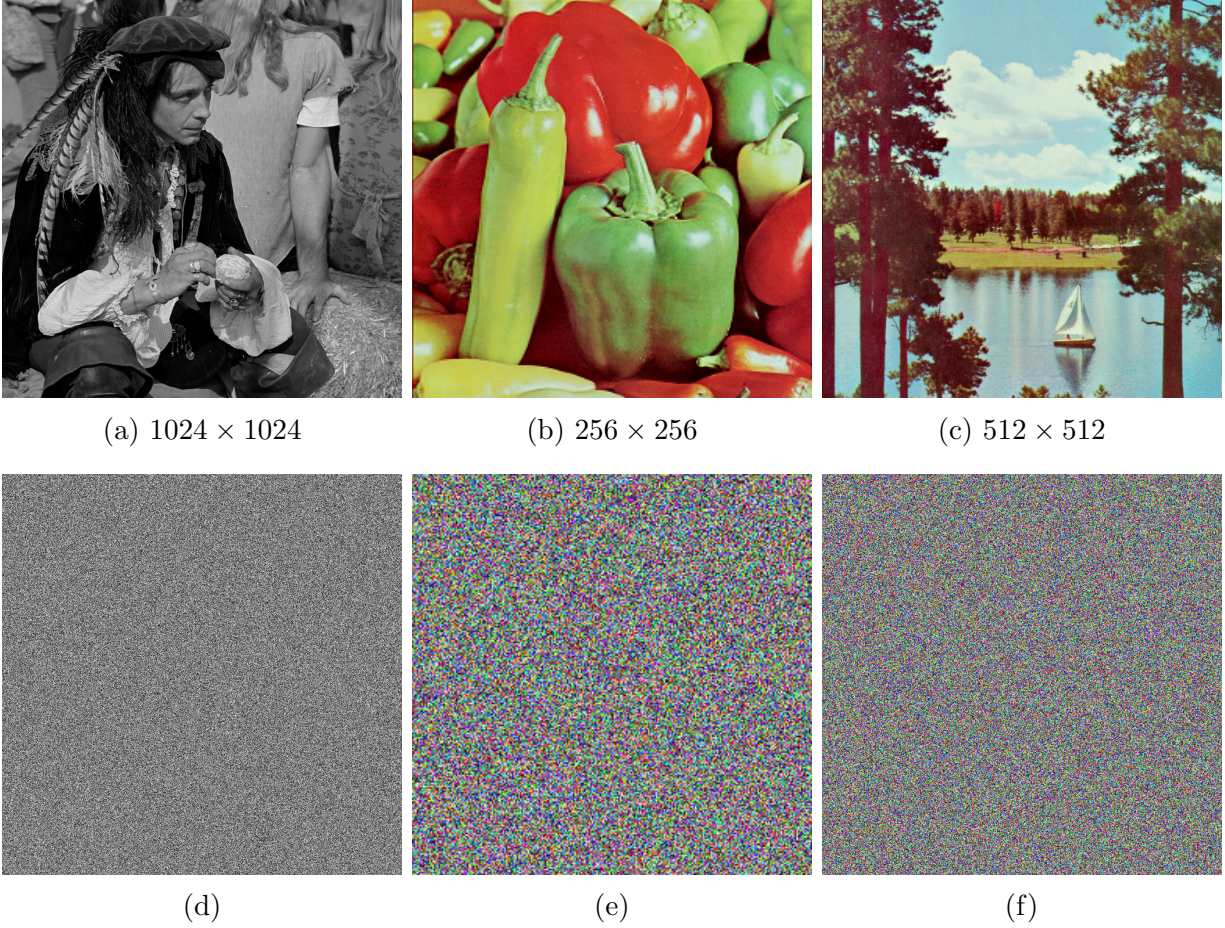


Figure 4.3: Sample images and their respective encrypted ciphers: **(a)** Male, **(b)** Peppers, **(c)** Sailboat, **(d)** Male cipher, **(e)** Peppers cipher, **(f)** Sailboat cipher.

correlation between the adjacent pixels, and thus the Ascon algorithm can resist statistical attacks.

### Correlation Coefficient between Plaintext and Cipher Images

The results for the vertical, horizontal, and diagonal correlation coefficients between the plaintext and cipher images of adjacent pixels for the Ascon algorithm are displayed in Table 4.6. The results indicate that the correlation values between the plaintext and the cipher images produced by the Ascon algorithm are close to 0, which means a substantial reduction in correlation.

Table 4.4: Vertical, horizontal, and diagonal correlation coefficients of adjacent pixels of encrypted images.

| Image     | Size and Color   | Correlation Coefficient |            |          |
|-----------|------------------|-------------------------|------------|----------|
|           |                  | Vertical                | Horizontal | Diagonal |
| Baboon    | 512 × 512 Color  | 0.0017                  | -0.0001    | -0.00121 |
| Cameraman | 256 × 256 Gray   | -0.0005                 | 0.0003     | -0.0017  |
| Clock     | 256 × 256 Gray   | -0.0008                 | 0.0013     | -0.0007  |
| Male      | 1024 × 1024 Gray | -0.0001                 | 0.0003     | -0.0013  |
| Peppers   | 256 × 256 Color  | 0.0008                  | 0.0021     | -0.0003  |
| Sailboat  | 512 × 512 Color  | 0.0017                  | -0.0011    | 0.0009   |
| Average   |                  | -0.0001                 | 0.0008     | -0.0007  |

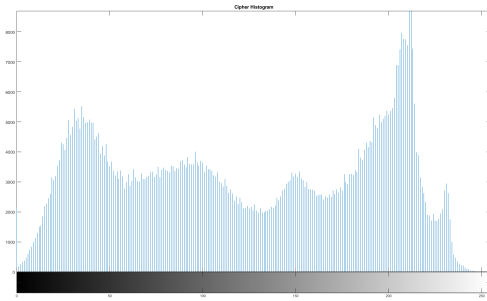
Table 4.5: Vertical, horizontal, and diagonal correlation coefficients of encrypted Baboon image for the Ascon algorithm compared to different encryption methods.

|       | Vertical       | Correlation Coefficient |               |
|-------|----------------|-------------------------|---------------|
|       |                | Horizontal              | Diagonal      |
| Ascon | 0.0017         | <b>-0.0001</b>          | -0.00121      |
| 100   | 0.0006         | 0.0005                  | <b>0.0007</b> |
| 8     | <b>-0.0001</b> | 0.0006                  | -0.0021       |
| 69    | -0.0086        | 0.0023                  | 0.0402        |
| 71    | -0.0036        | -0.0019                 | -0.0033       |
| 77    | -0.0004        | -0.0007                 | 0.0029        |

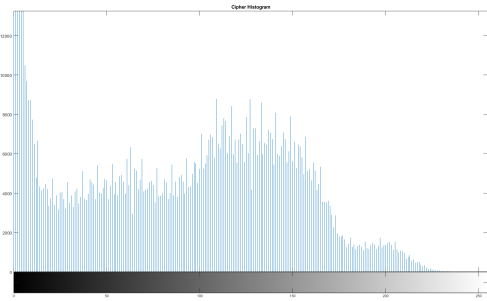
### 4.3.2 Histogram Analysis

Results indicate that the Ascon algorithm produces cipher images that are extremely close to being uniformly distributed. Figure 4.4 shows histograms of sample plaintext images and their respective ciphers. Results shows that Ascon is effective in hiding the plaintext image's information, and is deemed to be secure against histogram analysis attacks. A set of metrics are used to quantify histogram analysis, such as chi-square ( $\chi^2$ ), maximum deviation, irregular deviation, and deviation from uniform histogram. Each of the aforementioned metrics and their results are further discussed in the following subsections.

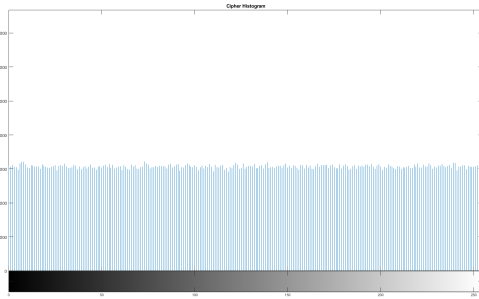




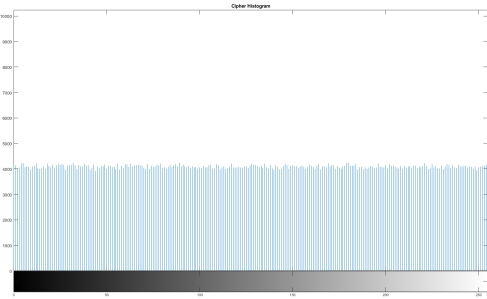
(a)



(b)



(c)



(d)

Figure 4.4: Histograms of images and their respective ciphers: (a) Sailboat histogram, (b) Male histogram, (c) Sailboat cipher histogram, (d) Male cipher histogram.

Table 4.6: Vertical, horizontal, and diagonal correlation coefficients of plaintext and cipher images.

| Image     | Size and Color   | Correlation Coefficient |            |          |
|-----------|------------------|-------------------------|------------|----------|
|           |                  | Vertical                | Horizontal | Diagonal |
| Baboon    | 512 × 512 Color  | 0.0001                  | 0.0002     | 0.0003   |
| Cameraman | 256 × 256 Gray   | -0.0004                 | -0.0005    | -0.0005  |
| Clock     | 256 × 256 Gray   | 0.0042                  | 0.0044     | 0.0043   |
| Male      | 1024 × 1024 Gray | 0.0008                  | 0.0007     | 0.0007   |
| Peppers   | 256 × 256 Color  | -0.0011                 | -0.0012    | 0.0012   |
| Sailboat  | 512 × 512 Color  | 0.0011                  | 0.0011     | 0.0011   |
| Average   |                  | 0.0007                  | 0.0008     | 0.0007   |

### Chi-Square ( $\chi^2$ )

Table 4.7 lists  $\chi^2$  test results. From the results in Table 4.7, it can be observed that the  $\chi^2$  values of images encrypted by Ascon are extremely low and are nearly close to a uniform distribution. Table 4.8 presents a comparison of Ascon algorithm to related image encryption algorithms. The results show that Ascon produces acceptable results; however, other algorithm designed specifically for image encryption produced better results for  $\chi^2$  metric.

Table 4.7: Chi-square test values for cipher images produced by Ascon algorithm.

| Image     | Size and Color   | Chi-square |
|-----------|------------------|------------|
| Baboon    | 512 × 512 Color  | 264.887    |
| Cameraman | 256 × 256 Gray   | 278.382    |
| Clock     | 256 × 256 Gray   | 249.199    |
| Male      | 1024 × 1024 Gray | 262.222    |
| Peppers   | 256 × 256 Color  | 263.618    |
| Sailboat  | 512 × 512 Color  | 256.235    |
| Average   |                  | 262.424    |

Table 4.8: Chi-square test values of Ascon algorithm compared to different encryption algorithms.

|            | Ascon   | 100            | 8       | 96      | 98      |
|------------|---------|----------------|---------|---------|---------|
| Chi-square | 262.424 | <b>228.447</b> | 273.271 | 252.000 | 265.000 |

## Maximum Deviation

Table 4.9 lists a comparison of  $D_{max}$  for cipher images encrypted by Ascon with other relevant encryption algorithms. From the results, it can be observed that the cipher images produced by Ascon algorithm are highly deviated from the original images. However, the performance of the algorithm by Alghamdi and Munir<sup>100</sup> is slightly better than Ascon algorithm for most cipher images and on average.

Table 4.9: Maximum deviation results of Ascon algorithm compared to related algorithms.

| Image     | Ascon        | 100           | 8      | 71     | 85    |
|-----------|--------------|---------------|--------|--------|-------|
| Baboon    | 261632       | <b>438546</b> | 363121 | 199158 | -     |
| Cameraman | <b>65403</b> | 65235         | 64382  | 64998  | 18007 |
| Clock     | <b>65397</b> | 62822         | -      | -      | 24823 |
| Peppers   | 165402       | <b>256837</b> | 209618 | 146408 | 22935 |
| Sailboat  | 261627       | <b>581918</b> | -      | -      | 25442 |
| Average   | 310985       | <b>319498</b> | 167482 | 102002 | 20109 |

## Irregular Deviation

Table 4.10 compares the  $D_{irregular}$  for Ascon with related image encryption algorithms. From the results, it can be observed that the Ascon  $D_{irregular}$  values are small, yet, the performance of the encryption algorithm by Alghamdi and Munir<sup>100</sup> is slightly better than Ascon for some cipher images as well as on average.

Table 4.10: Irregular deviation results of Ascon compared to other image encryption algorithms.

| Image     | Ascon        | 100          | 8     | 71     | 85           |
|-----------|--------------|--------------|-------|--------|--------------|
| Baboon    | <b>54733</b> | 59876        | 59921 | 80203  | -            |
| Cameraman | 39171        | <b>32111</b> | 32165 | 32706  | 39224        |
| Clock     | <b>26849</b> | 47060        | -     | -      | 27442        |
| Peppers   | 55134        | <b>23901</b> | 76075 | 84465  | 35088        |
| Sailboat  | 45563        | 72916        | -     | -      | <b>36226</b> |
| Average   | 134831       | <b>39311</b> | 91460 | 129253 | 40739        |

## Deviation from Uniform Histogram

Table 4.11 compares the  $D_{uniform}$  for Ascon and compares it with related image encryption algorithms. From the results, it can be seen that the Ascon cipher images'  $D_{uniform}$  values are small, yet, the performance of the encryption algorithm by Alghamdi and Munir<sup>100</sup> is slightly better for all tested cipher images and also on average.

Table 4.11: Deviation from uniform histogram of the Ascon algorithm compared to other encryption algorithms.

| Image     | Ascon  | 100           | 8             | 71     | 85     |
|-----------|--------|---------------|---------------|--------|--------|
| Baboon    | 0.0251 | <b>0.0242</b> | 0.0256        | 0.9990 | -      |
| Cameraman | 0.0521 | 0.0477        | <b>0.0305</b> | 0.9961 | 0.0942 |
| Clock     | 0.0498 | <b>0.0445</b> | -             | -      | 0.0949 |
| Peppers   | 0.0498 | 0.0479        | <b>0.0315</b> | 0.9990 | 0.0917 |
| Sailboat  | 0.0247 | <b>0.0241</b> | -             | -      | 0.0958 |
| Average   | 0.0358 | <b>0.0333</b> | 0.0382        | .9984  | 0.0940 |

### 4.3.3 Entropy Analysis

Table 4.12 lists the global and local entropy values for Ascon algorithm, and Table 4.13 compares Ascon's global entropy to related image encryption algorithms. From the results, it can be observed that the Ascon entropy values are extremely close to the ideal entropy. The global entropy of encryption algorithm by Alghamdi et. al<sup>8</sup> is better than Ascon's but the difference is extremely small.

### 4.3.4 Gray-Level Co-Occurrence Matrix (GLCM) Analysis

The GLCM test analyzes the spatial relationship of pixels and is used to determine the texture of an image produced by image encryption algorithms. The GLCM test comprises of a two-dimensional matrix of joint probabilities between pairs of pixels with specific values, and calculates the frequency of a pixel with gray-level value  $i$  occurring horizontally adjacent to a pixel with gray-level value  $j$ . Statistical measures, such as homogeneity, contrast, and

Table 4.12: Global and local entropy values of images encrypted by Ascon algorithm.

| Image     | Size and Color   | Entropy |        |
|-----------|------------------|---------|--------|
|           |                  | Global  | Local  |
| Baboon    | 512 × 512 Color  | 7.9997  | 7.7863 |
| Cameraman | 256 × 256 Gray   | 7.9969  | 7.7471 |
| Clock     | 256 × 256 Gray   | 7.9973  | 7.9732 |
| Male      | 1024 × 1024 Gray | 7.9998  | 7.9187 |
| Peppers   | 256 × 256 Color  | 7.9991  | 7.9892 |
| Sailboat  | 512 × 512 Color  | 7.9998  | 7.8540 |
| Average   |                  | 7.9988  | 7.8781 |

Table 4.13: Global entropy of cipher image Baboon of Ascon compared to different encryption algorithms.

| Entropy | Ascon  | 100    | 8             | 69     | 71     | 77     |
|---------|--------|--------|---------------|--------|--------|--------|
| Global  | 7.9988 | 7.9996 | <b>7.9997</b> | 7.9024 | 7.9992 | 7.9971 |

energy can be obtained from the GLCM to render information about the texture features of an image. In the following, we discuss contrast, energy, and homogeneity measures obtained from the GLCM.

### Contrast

Table 4.14 lists the contrast analysis results for plaintext images and their ciphers. The results show that cipher images produced by Ascon have high levels of contrast.

Table 4.14: Contrast analysis for plaintext and cipher images.

| Image     | Size and Color   | Plaintext Image | Cipher Image |
|-----------|------------------|-----------------|--------------|
| Baboon    | 512 × 512 Color  | 0.74257         | 10.5095      |
| Cameraman | 256 × 256 Gray   | 0.58716         | 10.5329      |
| Clock     | 256 × 256 Gray   | 0.36036         | 10.4855      |
| Male      | 1024 × 1024 Gray | 0.25280         | 10.5012      |
| Peppers   | 256 × 256 Color  | 0.42068         | 10.4763      |
| Sailboat  | 512 × 512 Color  | 0.41400         | 10.5003      |
| Average   |                  |                 | 10.5010      |

## Energy

Table 4.15 lists the energy analysis results for plaintext images and their ciphers. The results show that cipher images produced by Ascon have low levels of energy.

Table 4.15: Energy analysis for plaintext and cipher images.

| Image     | Size and Color          | Plaintext Image | Cipher Image |
|-----------|-------------------------|-----------------|--------------|
| Baboon    | $512 \times 512$ Color  | 0.06399         | 0.01563      |
| Cameraman | $256 \times 256$ Gray   | 0.18054         | 0.01564      |
| Clock     | $256 \times 256$ Gray   | 0.20371         | 0.01564      |
| Male      | $1024 \times 1024$ Gray | 0.11994         | 0.01563      |
| Peppers   | $256 \times 256$ Color  | 0.14053         | 0.01564      |
| Sailboat  | $512 \times 512$ Color  | 0.11551         | 0.01563      |
| Average   |                         |                 | 0.01563      |

## Homogeneity

Table 4.16 lists the homogeneity analysis results for plaintext images and their ciphers produced by Ascon. From the results, it can be seen that the Ascon produced low homogeneity values. Table 4.17 compares the Ascon's contrast, energy, and homogeneity average values to related image encryption algorithms. The results demonstrate that the Ascon produces excellent contrast values, but the contrast value in <sup>100</sup> is lower than Ascon by a small amount. Furthermore, the energy values in <sup>89</sup> are also slightly lower than those of Ascon's.

Table 4.16: Homogeneity analysis for plaintext and cipher images.

| Image     | Size and Color          | Plaintext Image | Cipher Image |
|-----------|-------------------------|-----------------|--------------|
| Baboon    | $512 \times 512$ Color  | 0.7653          | 0.3893       |
| Cameraman | $256 \times 256$ Gray   | 0.8953          | 0.3893       |
| Clock     | $256 \times 256$ Gray   | 0.9131          | 0.3885       |
| Male      | $1024 \times 1024$ Gray | 0.8979          | 0.3893       |
| Peppers   | $256 \times 256$ Color  | 0.8837          | 0.3900       |
| Sailboat  | $512 \times 512$ Color  | 0.8475          | 0.3894       |
| Average   |                         |                 | 0.3893       |

Table 4.17: Comparison of homogeneity, contrast, and energy values of cipher images of Ascon and related methods.

|             | <b>Ascon</b>  | <b>100</b>     | <b>8</b> | <b>89</b>      | <b>88</b> |
|-------------|---------------|----------------|----------|----------------|-----------|
| Contrast    | 10.5010       | <b>10.5357</b> | 10.5081  | 8.6448         | 8.3301    |
| Energy      | 0.01563       | 0.01563        | 0.01563  | <b>0.01561</b> | 0.01760   |
| Homogeneity | <b>0.3893</b> | 0.3895         | 0.3837   | 0.4110         | 0.4208    |

### 4.3.5 Encryption Quality

To quantify the quality of Ascon for image encryption, we have evaluated Ascon by a set of metrics, namely, mean square error (MSE), mean absolute error (MAE), and peak signal-to-noise ratio (PSNR). These tests compare the pixel values of plaintext images and their ciphers. The encryption algorithm is considered of a better quality when there is a substantial amount of change in pixel values between plaintext images and their ciphers. Each of these metrics is discussed further in the following.

#### Mean Square Error

Table 4.18 lists the Ascon MSE test results between plaintext images and their ciphers. From the results, it can be observed that Ascon image encryption yields high MSE values (i.e.,  $\geq 30$ ), and thus has high quality encryption. Table 4.19 compares Ascon MSE results to other image encryption algorithms. From the comparison results, it can be discerned that the cipher images produced by Ascon have great encryption quality; however, encryption algorithm in<sup>8</sup> performs slightly better than Ascon.

Table 4.18: MSE values of images encrypted by Ascon.

| <b>Image</b> | <b>Color and size</b>   | <b>MSE</b> |
|--------------|-------------------------|------------|
| Baboon       | 512 $\times$ 512 Color  | 46.0279    |
| Cameraman    | 256 $\times$ 256 Gray   | 35.0591    |
| Clock        | 256 $\times$ 256 Gray   | 32.0518    |
| Male         | 1024 $\times$ 1024 Gray | 47.6621    |
| Peppers      | 256 $\times$ 256 Color  | 26.8401    |
| Sailboat     | 512 $\times$ 512 Color  | 53.6175    |
| Average      |                         | 40.2097    |

Table 4.19: MSE values for images encrypted by Ascon compared to related methods.

|     | <b>Ascon</b> | <b>100</b> | <b>8</b>       | <b>71</b> | <b>80</b> |
|-----|--------------|------------|----------------|-----------|-----------|
| MSE | 40.2097      | 40.4568    | <b>42.3794</b> | 39.6794   | 33.4275   |

### Mean Absolute Error

Table 4.20 lists the Ascon MAE values test results between plaintext images and their ciphers. Results indicate that the Ascon algorithm yields high MAE values, and thus has high quality encryption. Table 4.21 compares Ascon MAE results to other image encryption algorithms. From the comparison results, it can be observed that the cipher images produced by Ascon have great encryption quality; however, encryption algorithm in<sup>100</sup> performs slightly better.

Table 4.20: MAE values of images encrypted by Ascon.

| <b>Image</b> | <b>Color and size</b> | <b>MAE</b> |
|--------------|-----------------------|------------|
| Baboon       | 512 × 512 Color       | 89.8333    |
| Cameraman    | 256 × 256 Gray        | 93.6492    |
| Clock        | 256 × 256 Gray        | 96.9981    |
| Male         | 1024 × 1024 Gray      | 97.4566    |
| Peppers      | 256 × 256 Color       | 96.9073    |
| Sailboat     | 512 × 512 Color       | 96.6205    |
| Average      |                       | 95.4566    |

Table 4.21: MAE values for images encrypted by Ascon compared to related encryption algorithms.

|     | <b>Ascon</b> | <b>100</b>     | <b>8</b> | <b>82</b> | <b>83</b> | <b>84</b> |
|-----|--------------|----------------|----------|-----------|-----------|-----------|
| MAE | 95.4566      | <b>95.5468</b> | 91.58    | 79.57     | 78.10     | 90        |

### Peak Signal-to-Noise Ratio

Table 4.22 lists the PSNR values of cipher images encrypted by Ascon. From the results, it can be seen that the Ascon algorithm has a low PSNR, therefore, Ascon produces good



quality image encryption. Table 4.23 compares the PSNR values of Ascon to other contemporary image encryption algorithms. From the results, it can be noticed that the cipher images produced by Ascon algorithm have better encryption quality than other compared encryption algorithms with respect to the PSNR metric.

Table 4.22: PSNR results of images encrypted by Ascon.

| Image     | Size and color          | PSNR   |
|-----------|-------------------------|--------|
| Baboon    | $512 \times 512$ Color  | 8.7899 |
| Cameraman | $256 \times 256$ Gray   | 8.3756 |
| Clock     | $256 \times 256$ Gray   | 7.2918 |
| Male      | $1024 \times 1024$ Gray | 8.1126 |
| Peppers   | $256 \times 256$ Color  | 8.1126 |
| Sailboat  | $512 \times 512$ Color  | 8.1802 |
| Average   |                         | 8.1262 |

Table 4.23: PSNR of the images encrypted by Ascon compared to other encryption algorithms.

|      | Ascon         | 100    | 8      | 71     | 82     | 53     |
|------|---------------|--------|--------|--------|--------|--------|
| PSNR | <b>8.1262</b> | 8.4569 | 8.4458 | 9.5424 | 9.0996 | 9.7936 |

### 4.3.6 Resistance Against Differential Attacks

Cryptanalysts utilize differential attacks to find a connection between a plaintext image and its cipher, by making small changes to the plaintext image and encrypting it with the same key. To evaluate the Ascon's ability to resist differential attacks and to assess its diffusion characteristics, we have tested Ascon using metrics such as avalanche effect, number of pixels change rate (NPCR), and unified average changing intensity (UACI). These tests assess an algorithm's resistance against differential attacks by encrypting a plaintext image two times. The first cipher image is obtained by encrypting the plaintext image, while the second cipher image is obtained by modifying a random pixel value in the original plaintext image and then encrypting the image. Each of these tests and their results are discussed in detail below.

## Avalanche Effect

Table 4.24 lists the  $MSE_{av}$  values of the Ascon algorithm. From the results, it can be noticed that the Ascon produces high  $MSE_{av}$  values (i.e.,  $\geq 50$ ), and thus cipher images produced by Ascon are considered to be safe against differential attacks.

Table 4.24: Avalanche effect  $MSE_{av}$  values of images encrypted by Ascon.

| Image     | Size and Color          | $MSE_{av}$ |
|-----------|-------------------------|------------|
| Baboon    | $512 \times 512$ Color  | 57.4127    |
| Cameraman | $256 \times 256$ Gray   | 57.8527    |
| Clock     | $256 \times 256$ Gray   | 56.6153    |
| Male      | $1024 \times 1024$ Gray | 58.8275    |
| Peppers   | $256 \times 256$ Color  | 57.7193    |
| Sailboat  | $512 \times 512$ Color  | 57.3573    |
| Average   |                         | 57.6308    |

## Number of Pixels Change Rate (NPCR)

Table 4.25 lists the NPCR values for Ascon. From the results, it can be noticed that the Ascon algorithm produces high NPCR values, therefore, cipher images produced by Ascon are considered to be safe against differential attacks. Table 4.26 compares the NPCR values of Ascon with related image encryption algorithms. Results in Table 4.26 indicate that the NPCR values of the Ascon algorithm are higher than other image encryption algorithms.

Table 4.25: NPCR values of images encrypted by Ascon.

| Image     | Size and Color          | NPCR    |
|-----------|-------------------------|---------|
| Baboon    | $512 \times 512$ Color  | 99.6249 |
| Cameraman | $256 \times 256$ Gray   | 99.6246 |
| Clock     | $256 \times 256$ Gray   | 99.6542 |
| Male      | $1024 \times 1024$ Gray | 99.6017 |
| Peppers   | $256 \times 256$ Color  | 99.6414 |
| Sailboat  | $512 \times 512$ Color  | 99.6137 |
| Average   |                         | 99.6267 |

Table 4.26: NPCR of Ascon compared with other encryption methods.

|      | <b>Ascon</b>   | <b>100</b> | <b>8</b> | <b>69</b> | <b>71</b> | <b>77</b> |
|------|----------------|------------|----------|-----------|-----------|-----------|
| NPCR | <b>99.6267</b> | 99.6197    | 99.6153  | 99.5893   | 99.6059   | 99.5743   |

### Unified Average Changing Intensity (UACI)

Table 4.27 lists the UACI values for Ascon. From the results, it can be seen that the Ascon algorithm produces high UACI values, therefore, cipher images produced by Ascon are considered to be resilient against differential attacks. Table 4.28 compares the UACI values of Ascon with other image encryption algorithms. Results in Table 4.28 indicate that the UACI values of the Ascon algorithm are better than other image encryption algorithms.

Table 4.27: UACI values of images encrypted by Ascon.

| <b>Image</b> | <b>Size and Color</b> | <b>UACI</b> |
|--------------|-----------------------|-------------|
| Baboon       | 512 × 512 Color       | 33.5860     |
| Cameraman    | 256 × 256 Gray        | 33.5687     |
| Clock        | 256 × 256 Gray        | 33.6325     |
| Male         | 1024 × 1024 Gray      | 33.4964     |
| Peppers      | 256 × 256 Color       | 33.4946     |
| Sailboat     | 512 × 512 Color       | 33.5701     |
| Average      |                       | 33.5581     |

Table 4.28: UACI of Ascon compared with other encryption methods.

|      | <b>Ascon</b>   | <b>100</b> | <b>8</b> | <b>69</b> | <b>71</b> | <b>77</b> |
|------|----------------|------------|----------|-----------|-----------|-----------|
| UACI | <b>33.5581</b> | 33.5504    | 33.4718  | 33.3730   | 33.4375   | 33.3941   |

### 4.3.7 Resistance to Noise and Data Loss

When images are sent through communication networks, they might be affected data loss or noise, potentially leading to the corruption of the cipher images. These damages could also be done by attackers trying to destroy the cipher images, and such attacks are known as noise and occlusion attacks. To evaluate the Ascon algorithm's resilience against occlusion attacks,

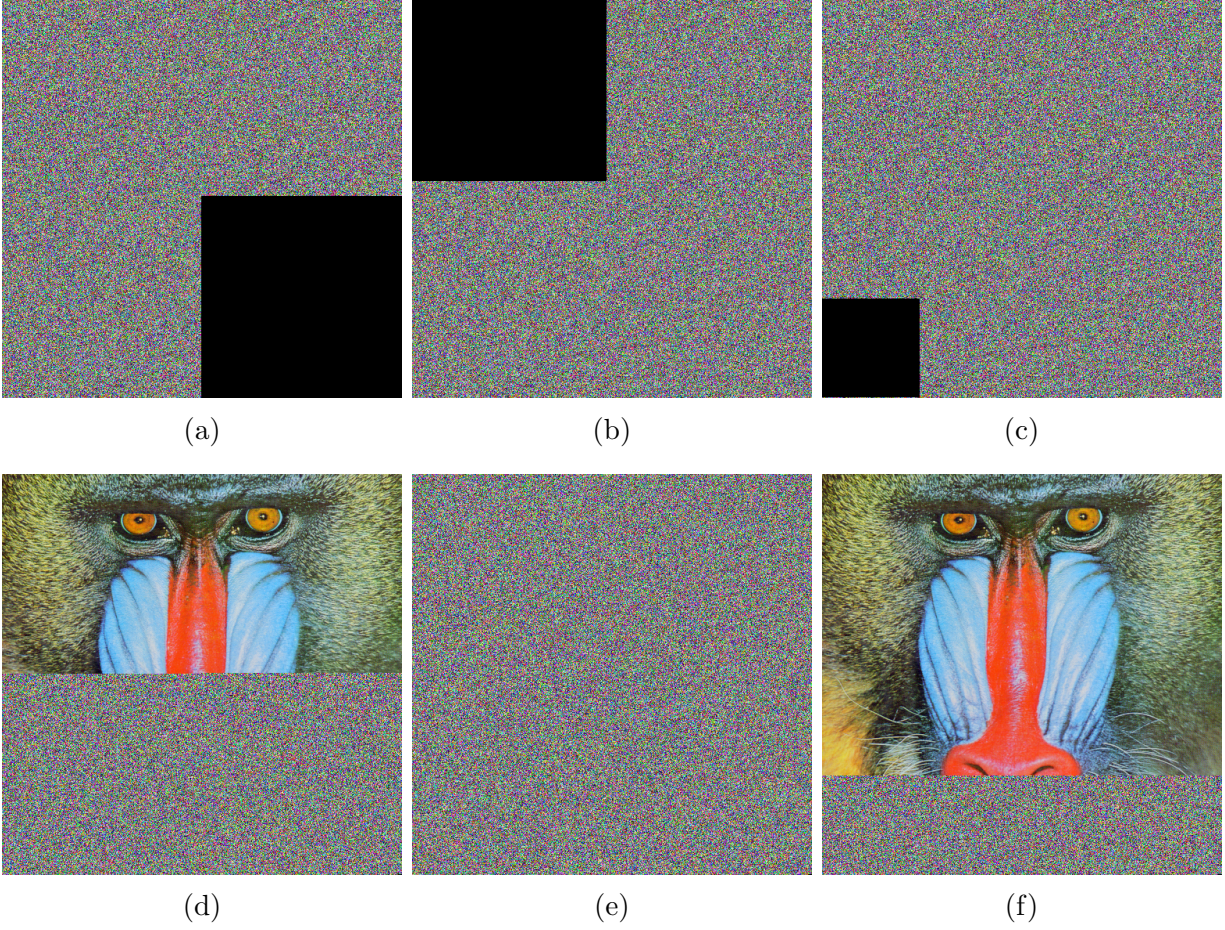


Figure 4.5: Results of occlusion attacks on the color image Baboon ( $512 \times 512$ ): (a–c) different severities of occlusion attacks, (d–f) corresponding decrypted images.

segments of the cipher image are cropped out at varying degrees of severity and the damaged ciphers are decrypted to try and retrieve the plaintext image. For resistance against noise attacks, salt and pepper noise is added to the cipher image with variable densities, and the damaged ciphers are decrypted to try and retrieve the plaintext image. Figure 4.5 illustrates the effects of occlusion attacks on a cipher image with different severities. Figure 4.6 illustrates the effect of salt and pepper noise with a density of 0.001 on a cipher image of Baboon. From the results, it can be seen that Ascon is not able to retrieve the plaintext successfully both for occlusion attacks as well as salt and pepper noise. Since the Ascon algorithm is based on sponge construction and each block of the image is affected by the previous state, it is expected that the Ascon will have difficulties decrypting damaged ciphers. To mitigate this problem, random row and column permutations could be implemented before running



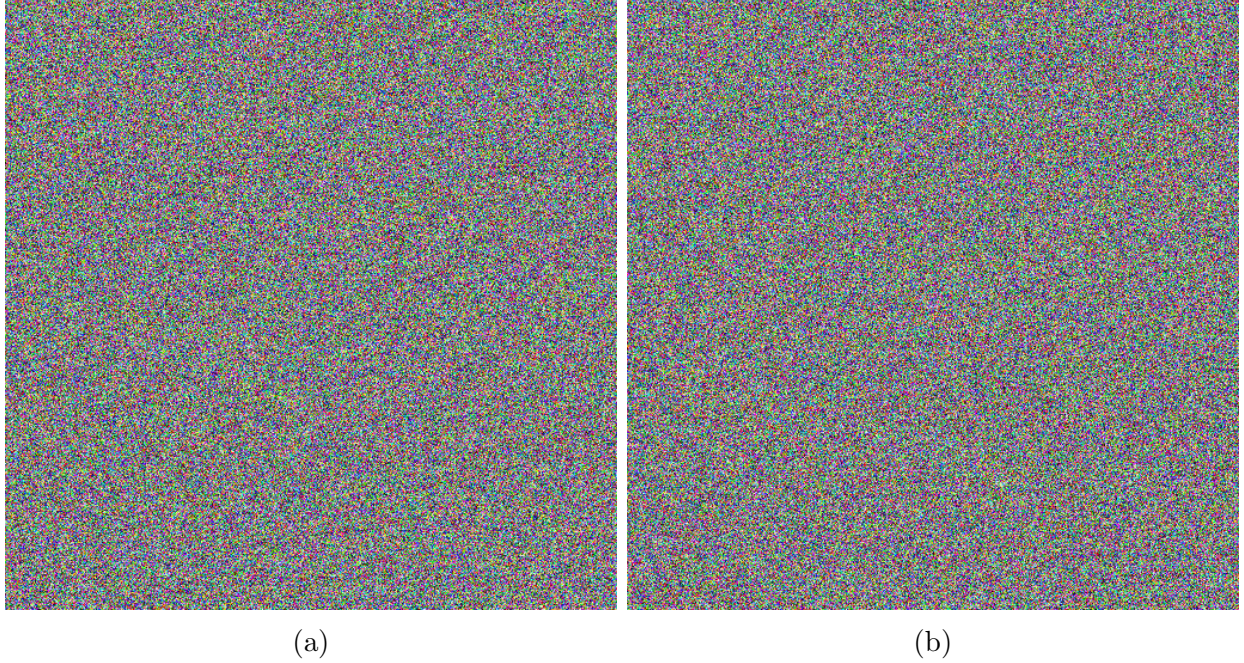


Figure 4.6: Results of salt and pepper noise attack on the color image Baboon ( $512 \times 512$ ): (a) salt and pepper noise with a density of 0.001, (b) corresponding decrypted image indicating that the image decryption failed.

the Ascon encryption.

### 4.3.8 Key Sensitivity

The Ascon algorithm's key sensitivity analysis is conducted by encrypting a plaintext image using two keys  $K_1$  and  $K_2$  that differ in only 1-bit. A key-sensitive algorithm should produce two completely different cipher images when encrypting the same plaintext image using  $K_1$  and  $K_2$ . Furthermore, the cipher image should not be able to be decrypted correctly using a key that has one bit altered from the original secret key. Figure 4.7 illustrates the results of key sensitivity tests for Ascon, from which it can be seen that Ascon is extremely sensitive to changes to the secret key. Figure 4.7 (b), (c), and (d) illustrate how Ascon produces two completely different ciphers for the keys  $K_1$  and  $K_2$ . Figure 4.7 (f), (g), and (h) illustrate how decrypting the cipher image with a key that differs in 1-bit from the original key that is used to encrypt the plaintext image, does not recover the plaintext image.



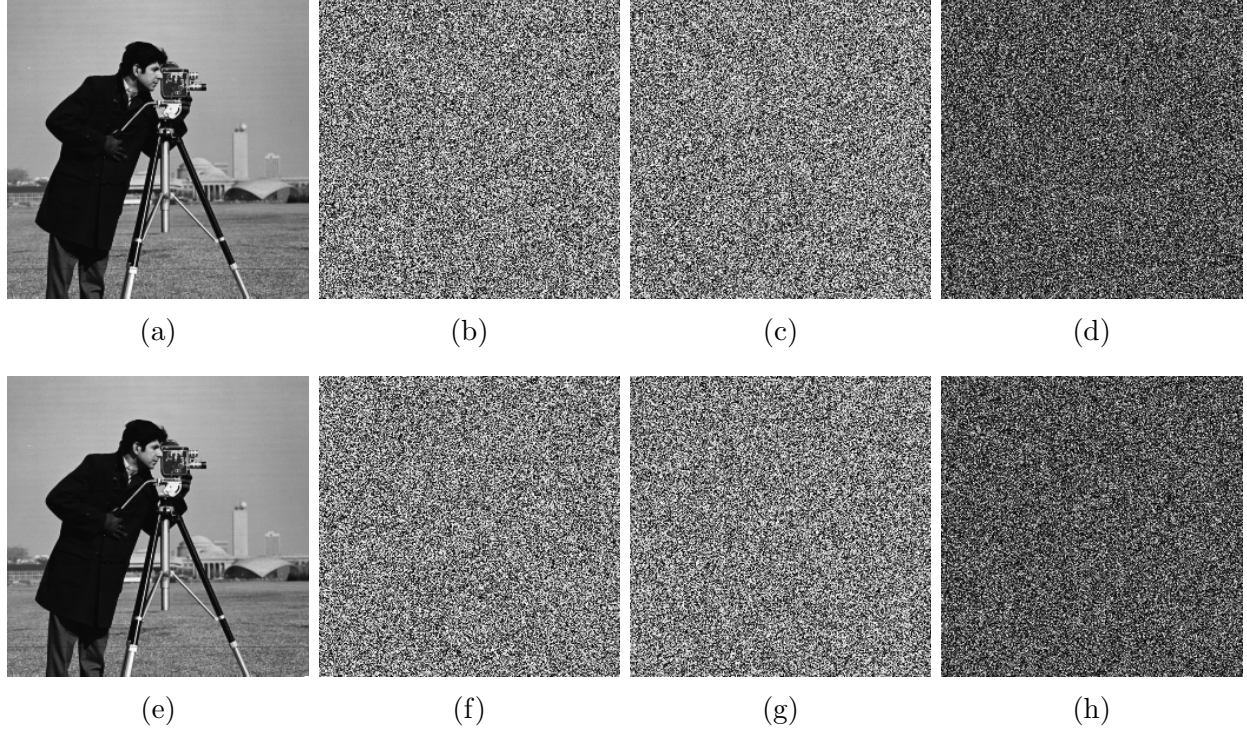


Figure 4.7: Results of key sensitivity tests on the color image Sailboat ( $512 \times 512$ ): (a) plaintext image, (b)  $C_1$  encrypted using key  $K_1$ , (c)  $C_2$  encrypted using key  $K_2$ , (d) difference of encrypted images:  $-(b) - (c)-$ , (e)  $C_1$  decrypted using key  $K_1$ , (f)  $C_1$  decrypted using key  $K_2$ , (g)  $C_2$  decrypted using key  $K_1$ , (h) difference of decrypted images:  $-(f) - (g)-$ .

#### 4.3.9 NIST SP 800-22 Test

Table 4.29 lists the results of the NIST SP 800-22 tests for the cipher of the image Male encrypted by Ascon. From the results, it can be discerned that the cipher image passes all the tests.

## 4.4 Conclusion

In this chapter, the new standard for lightweight cryptography applications, the Ascon algorithm, is tested for image encryption. This chapter analyzed and compared Ascon with various encryption algorithms in literature utilizing a comprehensive set of quality, efficiency, and security metrics. Experimental results demonstrated that the algorithm produces excellent results for correlation coefficient, chi-square, maximum deviation, irregular deviation,

Table 4.29: NIST SP 800-22 tests results for the encrypted Male image.

| Test                      |         | p-Value  | Passed |
|---------------------------|---------|----------|--------|
| Frequency                 |         | 0.328817 | ✓      |
| Block Frequency           |         | 0.122325 | ✓      |
| Cumulative Sums           | Reverse | 0.213309 | ✓      |
| Cumulative Sums           | Forward | 0.474986 | ✓      |
| Runs                      |         | 0.035174 | ✓      |
| Longest Run of Ones       |         | 0.017912 | ✓      |
| Rank                      |         | 0.242808 | ✓      |
| FFT                       |         | 0.202526 | ✓      |
| Non Overlapping Template  |         | 0.911413 | ✓      |
| Overlapping Template      |         | 0.739918 | ✓      |
| Universal                 |         | 0.066882 | ✓      |
| Approximate Entropy       |         | 0.534146 | ✓      |
| Random Excursions         |         | 0.742591 | ✓      |
| Random Excursions Variant |         | 0.728588 | ✓      |
| Serial                    | Test 1  | 0.678686 | ✓      |
| Serial                    | Test 2  | 0.350485 | ✓      |
| Linear Complexity         |         | 0.202268 | ✓      |

deviation from the uniform histogram, and entropy compared to related image encryption algorithms. Ascon’s evaluation of encryption quality and texture analysis using the gray-level co-occurrence matrix (GLCM) shows that Ascon produces great results that are better or similar to related image encryption algorithms. Furthermore, Ascon is resilient against differential attacks. The only tests that Ascon fails are the noise and occlusion attacks, which is due to the sponge construction of the algorithm. This problem can be solved by implementing random row and column permutations before the Ascon encryption.

# Chapter 5

## Conclusion

In this dissertation, several state-of-the-art image encryption techniques were classified and explored. Their the advantages, disadvantages and the applications of each of the image encryption classifications were explored. The dissertation proposed two novel algorithms designed to encrypt images with low complexity, high efficiency, and reliable security. Furthermore, a comprehensive set of security, quality, and efficiency evaluation metrics were introduced and utilized in comparison of the proposed image encryption algorithms with the contemporary image encryption algorithms on these evaluation metrics is conducted in this dissertation.

Chapter 2 proposed an efficient and secure lightweight chaos-based image encryption algorithm. The proposed algorithm utilizes logistic maps to produce the chaotic permutation matrices and the keys used for XOR encryption and decryption. An AES S-Box is used to substitute the image pixel's values to further randomize the produced cipher image. The proposed algorithm was analyzed and compared against other encryption algorithms using a variety of security, quality, and efficiency metrics. Experimental results revealed that the proposed algorithm has the least execution time as compared to other related encryption algorithms, where the proposed algorithm was up to  $15.33\times$  faster. Experimental results further showed that the proposed algorithm produced promising results in statistical and randomness analysis. Moreover, encryption quality tests, namely, mean square error, mean



absolute error, peak signal-to-noise ratio demonstrated that the proposed algorithm yields better quality encryption as compared to other related image encryption algorithms. The proposed algorithm also showed high resistance against differential attack when tested for avalanche effect, NPCR, and UACI. Furthermore, the proposed algorithm provides high resistance against noise and data loss attacks, where a plaintext image was recovered from decrypting a cipher image with up to 50% data loss and up to 50% of pixels affected by noise. NIST SP 800-22 test suite to test showed that the cipher image produced by the proposed algorithm passed all the tests, thus ensured the randomness characteristics of the cipher image.

Chapter 3 proposed secure multi-round image encryption algorithm based on nonlinear feedback shift registers. The proposed algorithm uses Trivium cipher to produce key-streams from an 80-bit round key and an 80-bit IV. Each key-stream is used once for each round, and used to perform pixel-based row permutation, pixel-based column permutation, and bit-level substitution using bitwise XOR on the plaintext image to produce the cipher image. The proposed algorithm was analyzed and compared with various encryption algorithms in literature utilizing a comprehensive set of quality, efficiency, and security metrics. The experimental results reveal that the proposed algorithm produces better results in terms of statistical and randomness analysis when compared to other contemporary image encryption algorithms. When tested for encryption quality, the proposed algorithm demonstrated excellent quality results similar to other contemporary image encryption methods. Additionally, experimental results reveal that the proposed algorithm was resilient against differential attacks and noise and occlusion attacks.

In chapter 4, the new standard for lightweight cryptography applications Ascon algorithm was tested for image encryption, analyzed and compared to various encryption algorithms in literature. The results demonstrated that the algorithm produced excellent results for statistical analysis compared to related image encryption algorithms. Ascon's quality of encryption evaluation and texture analysis showed that Ascon produced great results that are better or similar to related image encryption algorithms. Furthermore, Ascon was resilient against differential attacks. The only test that Ascon failed was the noise and occlusion

attacks due to the sponge construction of the algorithm.

Even though image encryption researchers utilize a comprehensive set of security, quality, and efficiency metrics to evaluate image encryption algorithms, there is a pressing need to establish rigorous security analysis and standardized evaluation metrics. Standardizing the evaluation metrics will not only better assess the resilience of image encryption methods, but will also guide researchers in testing their proposed algorithms. Researches also need to consider robustness of the image encryption algorithms against novel attacks, such as adversarial machine learning and quantum computing.

We plan to implement the proposed algorithms in hardware using a field-programmable gate array (FPGA). Furthermore, we plan to explore acceleration or parallelization of the proposed image encryption algorithm. Moreover, we plan to explore different implementations of Frequency domain-based, and Meaningful image encryption techniques that could provide higher security.

# Bibliography

- [1] Christophe De Canniere and Bart Preneel. Trivium specifications. *eSTREAM, ECRYPT Stream Cipher Project*, 2006, 2006. URL [http://www.ecrypt.eu.org/stream/p3ciphers/trivium/trivium\\_p3.pdf](http://www.ecrypt.eu.org/stream/p3ciphers/trivium/trivium_p3.pdf). Accessed: 2022-10-10.
- [2] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schl  ffer. Ascon v1. 2: Lightweight authenticated encryption and hashing. *Journal of Cryptology*, 34: 1–42, 2021.
- [3] Xing-yuan Wang, Feng Chen, and Tian Wang. A new compound mode of confusion and diffusion for block encryption of image based on chaos. *Communications in Nonlinear Science and Numerical Simulation*, 15(9):2479–2485, 2010.
- [4] Morris Dworkin, Elaine Barker, James Nechvatal, James Foti, Lawrence Bassham, E. Roback, and James Dray. Advanced encryption standard (aes), 2001-11-26 2001.
- [5] Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, and Niels Ferguson. Twofish: A 128-bit block cipher. *NIST AES Proposal*, 15(1):23–91, 1998.
- [6] Daniel J Bernstein et al. Chacha, a variant of salsa20. In *Workshop record of SASC*, volume 8, pages 3–5. Citeseer, 2008.
- [7] Tahir Sajjad Ali and Rashid Ali. A new chaos based color image encryption algorithm using permutation substitution and boolean operation. *Multimedia Tools and Applications*, 79(27-28):19853–19873, 2020.
- [8] Yousef Alghamdi, Arslan Munir, and Jawad Ahmad. A lightweight image encryption algorithm based on chaotic map and random substitution. *Entropy*, 24(10):1344, Sep 2022. ISSN 1099-4300. doi: 10.3390/e24101344. URL <http://dx.doi.org/10.3390/e24101344>.

- [9] Yuqin Luo, Jin Yu, Wenrui Lai, and Lingfeng Liu. A novel chaotic image encryption algorithm based on improved baker map and logistic map. *Multimedia Tools and Applications*, 78:22023–22043, 2019.
- [10] Bin Zhang, Bahbib Rahmatullah, Shir Li Wang, and Zhaoyan Liu. A plain-image correlative semi-selective medical image encryption algorithm using enhanced 2d-logistic map. *Multimedia Tools and Applications*, pages 1–28, 2022.
- [11] Ibrahim F Elashry, Walid El-Shafai, Emad S Hasan, S El-Rabaie, Alaa M Abbas, Fathi E Abd El-Samie, Hala S El-sayed, and Osama S Faragallah. Efficient chaotic-based image cryptosystem with different modes of operation. *Multimedia Tools and Applications*, 79:20665–20687, 2020.
- [12] Bhaskar Mondal, Prabhakar Kumar, and Shrey Singh. A chaotic permutation and diffusion based image encryption algorithm for secure communications. *Multimedia Tools and Applications*, 77:31177–31198, 2018.
- [13] Eko Hari Rachmawanto, Ignatius Moses Setiadi De Rosal, Christy Atika Sari, Heru Agus Santoso, Fauzi Adi Rafrastara, and Edi Sugiarto. Block-based arnold chaotic map for image encryption. In *2019 International Conference on Information and Communications Technology (ICOIACT)*, pages 174–178. IEEE, 2019.
- [14] Mohamed A Wahby Shalaby, Marwa T Saleh, and Hesham N Elmahdy. Enhanced arnold’s cat map-aes encryption technique for medical images. In *2020 2nd novel intelligent and leading emerging sciences conference (NILES)*, pages 288–295. IEEE, 2020.
- [15] Chunhu Li, Guangchun Luo, Ke Qin, and Chunbao Li. An image encryption scheme based on chaotic tent map. *Nonlinear Dynamics*, 87(1):127–133, 01 2017.
- [16] CGM Vishwas and R Sanjeev Kunte. An image cryptosystem based on tent map. In *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pages 1069–1073. IEEE, 2020.

- [17] Xiaohong Gao. Image encryption algorithm based on 2d hyperchaotic map. *Optics & Laser Technology*, 142:107252, 2021.
- [18] Yueping Li, Chunhua Wang, and Hua Chen. A hyper-chaos-based image encryption algorithm using pixel-level permutation and bit-level permutation. *Optics and Lasers in Engineering*, 90:238–246, 2017.
- [19] SC Phatak and S Suresh Rao. Logistic map: A possible random-number generator. *Physical review E*, 51(4):3670, 1995.
- [20] S Rohith, L Jahnavi, SC Bhuvaneshwari, S Supreeth, and BK Sujatha. Image encryption and decryption using key sequence of triple logistic map for medical applications. In *2020 Third International Conference on Advances in Electronics, Computers and Communications (ICAECC)*, pages 1–5. IEEE, 2020.
- [21] Vladimir Igorevich Arnold and André Avez. *Ergodic problems of classical mechanics*, volume 9. Benjamin, 1968.
- [22] Yifeng Zheng and Jianxiu Jin. A novel image encryption scheme based on h  non map and compound spatiotemporal chaos. *Multimedia Tools and Applications*, 74: 7803–7820, 2015.
- [23] Diya Achu Pradeep, A Harsha, and Jaison Jacob. Image encryption using chaotic map and related analysis. In *2021 International Conference on Advances in Computing and Communications (ICACC)*, pages 1–5. IEEE, 2021.
- [24] Khalid A Hussein, Sawsen A Mahmood, and Mazin A Abbass. A new permutation-substitution scheme based on henon chaotic map for image encryption. In *2019 2nd Scientific Conference of Computer Sciences (SCCS)*, pages 63–68. IEEE, 2019.
- [25] Radu Boriga, Ana Cristina D  sc  lescu, and Iustin Priescu. A new hyperchaotic map and its application in an image encryption scheme. *Signal Processing: Image Communication*, 29(8):887–901, 2014.

- [26] Milan Vispoel, AJ Daly, and JM Baetens. Lyapunov exponents of multi-state cellular automata. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 33(4):043108, 2023.
- [27] Bhat Jasra and Ayaz Hassan Moon. Color image encryption and authentication using dynamic dna encoding and hyper chaotic system. *Expert Systems with Applications*, 206:117861, 2022.
- [28] Priya Ramasamy, Vidhyapriya Ranganathan, Seifedine Kadry, Robertas Damaševičius, and Tomas Blažauskas. An image encryption scheme based on block scrambling, modified zigzag transformation and key generation using enhanced logistic—tent map. *Entropy*, 21(7):656, 2019.
- [29] Kurunandan Jain, Aravind Aji, and Prabhakar Krishnan. Medical image encryption scheme using multiple chaotic maps. *Pattern Recognition Letters*, 152:356–364, 2021.
- [30] Tsu-Yang Wu, Xiaoning Fan, King-Hang Wang, Chin-Feng Lai, Naixue Xiong, and Jimmy Ming-Tai Wu. A dna computation-based image encryption scheme for cloud cctv systems. *IEEE Access*, 7:181434–181443, 2019.
- [31] Kaishi Li and Zhen Su. Research on an image high intensive encryption way based on the chaos theory and dna coding. In *2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pages 1–7. IEEE, 2019.
- [32] Xue Li and Xiaohui Li. A novel block image encryption algorithm based on dna dynamic encoding and chaotic system. In *2019 IEEE 4th international conference on signal and image processing (ICSIP)*, pages 901–906. IEEE, 2019.
- [33] Wei Feng, Yigang He, Hongmin Li, and Chunlai Li. A plain-image-related chaotic image encryption algorithm based on dna sequence operation and discrete logarithm. *IEEE Access*, 7:181589–181609, 2019.

- [34] Zhenlong Man, Jinqing Li, Xiaoqiang Di, Yaohui Sheng, and Zefei Liu. Double image encryption algorithm based on neural network and chaos. *Chaos, Solitons & Fractals*, 152:111318, 2021.
- [35] Feifei Yang, Jun Mou, Yinghong Cao, and Ran Chu. An image encryption algorithm based on bp neural network and hyperchaotic system. *China Communications*, 17(5): 21–28, 2020.
- [36] Baoru Han, Yuanyuan Jia, Guo Huang, and Lisha Cai. A medical image encryption algorithm based on hermite chaotic neural network. In *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, volume 1, pages 2644–2648. IEEE, 2020.
- [37] Xing-Yuan Wang and Zhi-Ming Li. A color image encryption algorithm based on hopfield chaotic neural network. *Optics and Lasers in Engineering*, 115:107–118, 2019.
- [38] Lina Ding and Qun Ding. A novel image encryption scheme based on 2d fractional chaotic map, dwf and 4d hyper-chaos. *Electronics*, 9(8):1280, 2020.
- [39] Mengmeng Guan, Xuelin Yang, and Weisheng Hu. Chaotic image encryption algorithm using frequency-domain dna encoding. *IET image processing*, 13(9):1535–1539, 2019.
- [40] Yong Chen, Shucui Xie, and Jianzhong Zhang. A hybrid domain image encryption algorithm based on improved henon map. *Entropy*, 24(2):287, 2022.
- [41] Yuqiang Dou and Ming Li. An image encryption algorithm based on compressive sensing and m sequence. *IEEE Access*, 8:220646–220657, 2020.
- [42] Miao Zhang, Xiao-Jun Tong, Jie Liu, Zhu Wang, Jinlong Liu, Baolong Liu, and Jing Ma. Image compression and encryption scheme based on compressive sensing and fourier transform. *IEEE Access*, 8:40838–40849, 2020.
- [43] Xingyuan Wang and Yining Su. Image encryption based on compressed sensing and dna encoding. *Signal Processing: Image Communication*, 95:116246, 2021.



- [44] Ping Ping, Jie Fu, Yingchi Mao, Feng Xu, and Jerry Gao. Meaningful encryption: generating visually meaningful encrypted images by compressive sensing and reversible color transformation. *IEEE access*, 7:170168–170184, 2019.
- [45] Xingyuan Wang, Cheng Liu, and Donghua Jiang. Visually meaningful image encryption scheme based on new-designed chaotic map and random scrambling diffusion strategy. *Chaos, Solitons & Fractals*, 164:112625, 2022.
- [46] Donghua Jiang, Lidong Liu, Liya Zhu, Xingyuan Wang, Xianwei Rong, and Hongxiang Chai. Adaptive embedding: A novel meaningful image encryption scheme based on parallel compressive sensing and slant transform. *Signal Processing*, 188:108220, 2021.
- [47] Yu-Guang Yang, Bao-Pu Wang, Yong-Li Yang, Yi-Hua Zhou, Wei-Min Shi, and Xin Liao. Visually meaningful image encryption based on universal embedding model. *Information Sciences*, 562:304–324, 2021.
- [48] Shang Ma, Yan Zhang, Zeguo Yang, Jianhao Hu, and Xin Lei. A new plaintext-related image encryption scheme based on chaotic sequence. *IEEE Access*, 7:30344–30360, 2019.
- [49] Jan Sher Khan, Atique ur Rehman, Jawad Ahmad, and Zeeshan Habib. A new chaos-based secure image encryption scheme using multiple substitution boxes. In *2015 Conference on Information Assurance and Cyber Security (CIACS)*, pages 16–21, 2015. doi: 10.1109/CIACS.2015.7395561.
- [50] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.
- [51] Yue Wu, Yicong Zhou, George Saveriades, Sos Agaian, Joseph P. Noonan, and Premkumar Natarajan. Local shannon entropy measure with statistical tests for image randomness. *Information Sciences*, 222:323–342, 2013. ISSN 0020-0255. doi: <https://doi.org/10.1016/j.ins.2012.07.049>. URL <https://www.sciencedirect.com/>

[science/article/pii/S002002551200521X](https://doi.org/10.1109/ACCESS.2021.3025241). Including Special Section on New Trends in Ambient Intelligence and Bio-inspired Systems.

- [52] Yuwen Sha, Yinghong Cao, Huizhen Yan, Xinyu Gao, and Jun Mou. An image encryption scheme based on iavl permutation scheme and dna operations. *IEEE Access*, 9:96321–96336, 2021.
- [53] Abdullah Qayyum, Jawad Ahmad, Wadii Boulila, Saeed Rubaiee, Fawad Masood, Fawad Khan, William J Buchanan, et al. Chaos-based confusion and diffusion of image pixels using dynamic substitution. *IEEE Access*, 8:140876–140895, 2020.
- [54] Nadeem Iqbal, Muhammad Hanif, Sagheer Abbas, Muhammad Adnan Khan, Sultan H. Almotiri, and Mohammed A. Al Ghamdi. Dna strands level scrambling based color image encryption scheme. *IEEE Access*, 8:178167–178182, 2020. doi: 10.1109/ACCESS.2020.3025241.
- [55] Sarita D Sanap and Vijayshree More. Performance analysis of encryption techniques based on avalanche effect and strict avalanche criterion. In *2021 3rd International Conference on Signal Processing and Communication (ICPSC)*, pages 676–679. IEEE, 2021.
- [56] Fei Yu, Hui Shen, Qiulin Yu, Xinxin Kong, Pradip Kumar Sharma, and Shuo Cai. Privacy protection of medical data based on multi-scroll memristive hopfield neural network. *IEEE Transactions on Network Science and Engineering*, 10(2):845–858, 2023. doi: 10.1109/TNSE.2022.3223930.
- [57] Yue Wu, Joseph P Noonan, Sos Agaian, et al. Npcr and uaci randomness tests for image encryption. *Cyber journals: multidisciplinary journals in science and technology, Journal of Selected Areas in Telecommunications (JSAT)*, 1(2):31–38, 2011.
- [58] Arslan Shafique, Jameel Ahmed, Mujeeb Ur Rehman, and Mohammad Mazyad Hazazi. Noise-resistant image encryption scheme for medical images in the chaos and

- wavelet domain. *IEEE Access*, 9:59108–59130, 2021. doi: 10.1109/ACCESS.2021.3071535.
- [59] Jie Zhou, Nan-Run Zhou, and Li-Hua Gong. Fast color image encryption scheme based on 3d orthogonal latin squares and matching matrix. *Optics & Laser Technology*, 131: 106437, 2020.
- [60] Gonzalo Alvarez and Shujun Li. Some basic cryptographic requirements for chaos-based cryptosystems. *International journal of bifurcation and chaos*, 16(08):2129–2151, 2006.
- [61] Varsha Himthani, Vijaypal Singh Dhaka, Manjit Kaur, Dilbag Singh, and Heung-No Lee. Systematic survey on visually meaningful image encryption techniques. *IEEE Access*, 10:98360–98373, 2022. doi: 10.1109/ACCESS.2022.3203173.
- [62] Lawrence Bassham, Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Stefan Leigh, M Levenson, M Vangel, Nathanael Heckert, and D Banks. A statistical test suite for random and pseudorandom number generators for cryptographic applications, 2010-09-16 2010. URL [https://tsapps.nist.gov/publication/get\\_pdf.cfm?pub\\_id=906762](https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=906762).
- [63] Fawad Ahmed, MY Siyal, and Vali Uddin Abbas. A perceptually scalable and jpeg compression tolerant image encryption scheme. In *2010 Fourth Pacific-Rim Symposium on Image and Video Technology*, pages 232–238. IEEE, 2010.
- [64] Arslan Munir, Prasanna Kansakar, and Samee U. Khan. IFCIoT: Integrated Fog Cloud IoT: A novel architectural paradigm for the future Internet of Things. *IEEE Consumer Electronics Magazine*, 6(3):74–82, 2017. ISSN 2162-2248.
- [65] Arslan Munir, Erik Blasch, Jisu Kwon, Joonho Kong, and Alexander Aved. Artificial intelligence and data fusion at the edge. *IEEE Aerospace and Electronic Systems Magazine*, 36(7):62—78, July 2021.

- [66] Philip P Dang and Paul M Chau. Image encryption for secure internet multimedia applications. *IEEE Transactions on consumer electronics*, 46(3):395–403, 2000.
- [67] Vinod Patidar, NK Pareek, and KK Sud. A new substitution–diffusion based image cipher using chaotic standard and logistic maps. *Communications in Nonlinear Science and Numerical Simulation*, 14(7):3056–3075, 2009.
- [68] Kalyani Mali, Shouvik Chakraborty, and Mousomi Roy. A study on statistical analysis and security evaluation parameters in image encryption. *entropy*, 34:36, 2015.
- [69] Zhongyun Hua, Yicong Zhou, Chi-Man Pun, and CL Philip Chen. 2d sine logistic modulation map for image encryption. *Information Sciences*, 297:80–94, 2015.
- [70] Ahmad Alanezi, Bassem Abd-El-Atty, Hoshang Kolivand, Abd El-Latif, A Ahmed, Abd El-Rahiem, Syam Sankar, Hany S Khalifa, et al. Securing digital images through simple permutation-substitution mechanism in cloud-based smart city environment. *Security and Communication Networks*, 2021, 2021.
- [71] Jameel Arif, Muazzam A Khan, Baraq Ghaleb, Jawad Ahmad, Arslan Munir, Umer Rashid, and Ahmed Al-Dubai. A novel chaotic permutation–substitution image encryption scheme based on logistic map and random substitution. *IEEE Access*, 2022.
- [72] Xingyuan Wang, Xiaoqiang Zhu, and Yingqian Zhang. An image encryption algorithm based on josephus traversing and mixed chaotic map. *IEEE Access*, 6:23733–23746, 2018.
- [73] Steven H Strogatz. *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. CRC press, 2018.
- [74] Quynh Dang. Secure hash standard, 2015-08-04 2015.
- [75] Usc-sipi image database. volume 3: Miscellaneous. URL <https://sipi.usc.edu/database/database.php?volume=misc>.

- [76] Arslan Munir, Ann Gordon-Ross, Susan Lysecky, and Roman Lysecky. A Lightweight Dynamic Optimization Methodology and Application Metrics Estimation Model for Wireless Sensor Networks. *Elsevier Sustainable Computing: Informatics and Systems*, 3(2):94–108, June 2013.
- [77] Qing Lu, Congxu Zhu, and Xiaoheng Deng. An efficient image encryption scheme based on the lss chaotic map and single s-box. *IEEE Access*, 8:25664–25678, 2020.
- [78] Guanrong Chen, Yaobin Mao, and Charles K Chui. A symmetric image encryption scheme based on 3d chaotic cat maps. *Chaos, Solitons & Fractals*, 21(3):749–761, 2004.
- [79] Lini Abraham and Neenu Daniel. Secure image encryption algorithms: A review. *International journal of scientific & technology research*, 2(4):186–189, 2013.
- [80] Jawad Ahmad and Fawad Ahmed. Efficiency analysis and security evaluation of image encryption schemes. *computing*, 23:25, 2010.
- [81] Jansher Khan, Jawad Ahmad, and Seong Oun Hwang. An efficient image encryption scheme based on: Henon map, skew tent map and s-box. In *2015 6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO)*, pages 1–6. IEEE, 2015.
- [82] Muhammad Samiullah, Waqar Aslam, Hira Nazir, M Ikramullah Lali, Basit Shahzad, Muhammad Rafiq Mufti, and Humaira Afzal. An image encryption scheme based on dna computing and multiple chaotic systems. *IEEE Access*, 8:25650–25663, 2020.
- [83] Wassim Alexan, Mohamed ElBeltagy, and Amr Aboshousha. Rgb image encryption through cellular automata, s-box and the lorenz system. *Symmetry*, 14(3):443, 2022.
- [84] Majid Khan, Lalsaid Khan, Mohammad Mazyad Hazzazi, Sajjad Shaukat Jamal, and Iqtadar Hussain. Image encryption scheme for multi-focus images for visual sensors network. *Multimedia Tools and Applications*, 81(12):16353–16370, 2022.

- [85] Akram Belazi, Ahmed A Abd El-Latif, and Safya Belghith. A novel image encryption scheme based on substitution-permutation network and chaos. *Signal Processing*, 128: 155–170, 2016.
- [86] C Lakshmi, Karuppusamy Thenmozhi, John Bosco Balaguru Rayappan, and Rengarajan Amirtharajan. Hopfield attractor-trusted neural network: an attack-resistant image encryption. *Neural Computing and Applications*, 32(15):11477–11489, 2020.
- [87] Guodong Ye, Kaixin Jiao, Chen Pan, and Xiaoling Huang. An effective framework for chaotic image encryption based on 3d logistic map. *Security and Communication Networks*, 2018, 2018.
- [88] Amir Anees, Adil Masood Siddiqui, and Fawad Ahmed. Chaotic substitution for highly autocorrelated data in encryption algorithm. *Communications in Nonlinear Science and Numerical Simulation*, 19(9):3106–3118, 2014.
- [89] Fadia Ali Khan, Jameel Ahmed, Jan Sher Khan, Jawad Ahmad, and Muazzam A Khan. A novel image encryption based on lorenz equation, gingerbreadman chaotic map and s 8 permutation. *Journal of Intelligent & Fuzzy Systems*, 33(6):3753–3765, 2017.
- [90] Fawad Masood, Maha Driss, Wadii Boulila, Jawad Ahmad, Sadaqat Ur Rehman, Sana Ullah Jan, Abdullah Qayyum, and William J Buchanan. A lightweight chaos-based medical image encryption scheme using random shuffling and xor operations. *Wireless Personal Communications*, pages 1–28, 2021.
- [91] Lawrence Bassham, Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Stefan Leigh, M Levenson, M Vangel, Nathanael Heckert, and D Banks. A statistical test suite for random and pseudorandom number generators for cryptographic applications, 2010-09-16 2010. URL [https://tsapps.nist.gov/publication/get\\_pdf.cfm?pub\\_id=906762](https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=906762).
- [92] FIPS Pub. Data encryption standard (des). *FIPS PUB*, pages 46–3, 1999.

- [93] Mohamed Zakariya Talhaoui and Xingyuan Wang. A new fractional one dimensional chaotic map and its application in high-speed image encryption. *Information Sciences*, 550:13–26, 2021.
- [94] Yabin Zhang, Li Zhang, Zhi Zhong, Lei Yu, Mingguang Shan, and Yigui Zhao. Hyperchaotic image encryption using phase-truncated fractional fourier transform and dna-level operation. *Optics and Lasers in Engineering*, 143:106626, 2021.
- [95] Machbah Uddin, Farah Jahan, Mohammad Khairul Islam, and Md Rakib Hassan. A novel dna-based key scrambling technique for image encryption. *Complex & Intelligent Systems*, 7(6):3241–3258, 2021.
- [96] Di Xiao, Ayesha Kulsoom, Muntazim Abbas Hashmi, Syed Ali Abbas, et al. Block mode image encryption technique using two-fold operations based on chaos, md5 and dna rules. *Multimedia Tools and Applications*, 78(7):9355–9382, 2019.
- [97] M. A. Qureshi and A. Munir. PUF-RAKE: A PUF-based robust and lightweight authentication and key establishment protocol. *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 19(4):2457–2475, 2022.
- [98] Shahram Etemadi Borujeni and Mohammad Eshghi. Chaotic image encryption system using phase-magnitude transformation and pixel substitution. *Telecommunication Systems*, 52(2):525–537, 2013.
- [99] Seyed Shahabeddin Moafimadani, Yucheng Chen, and Chunming Tang. A new algorithm for medical color images encryption using chaotic systems. *Entropy*, 21(6), 2019. ISSN 1099-4300. doi: 10.3390/e21060577. URL <https://www.mdpi.com/1099-4300/21/6/577>.
- [100] Yousef Alghamdi and Arslan Munir. An image encryption algorithm based on trivium cipher and random substitution. *SN Computer Science*, 4(6):713, 2023.