

Image Features Hiding by Permutation of Pixels Locations

Mohammed Abdul-Hameed¹, Hamdy El-Metwally², Abdel Rahman Karawia³

^{1,2,3}Mathematics Department, Faculty of Science, Mansoura University, Mansoura 35516, Egypt

¹ College of Basic Education University of Kufa- As well.

mohammeda.alkufi@uokufa.edu.iq

helmetwally@mans.edu.eg

abibka@mans.edu.eg

ARTICLE INFO

ABSTRACT

Received: 05 Oct 2024

Revised: 07 Dec 2024

Accepted: 21 Dec 2024

After completing a scientific research project on image encryption using third-order differential equations, we will focus on designing a new algorithm to conceal the features of the image by exchanging pixel positions in a calculated, precise, and reversible mathematical manner. The exchange will occur in five stages, where the first four stages involve flipping the pixels in a way that resembles paper flipping, causing the pixels to interleave, making their features almost 90% disappear. It is quite natural to significantly increase the disappearance ratio of the image features, reaching up to 99%, by increasing the stages of pixel flipping in different and interleaved positions. However, this is not necessary as we will make the fifth stage of pixel position swapping using a chaotic map, which has the great ability to completely obscure the image features to a ratio of 100%. We applied this algorithm to several images, and the results were excellent. It is worth mentioning that, despite completely hiding the image features, this algorithm is not considered an encryption algorithm because a crucial aspect of encryption, namely the encryption key, is not present. However, it serves as a prelude to the encryption process, we have accomplished through our previous research, where the encryption here is more secure from a security perspective. Additionally, we note that it is possible to enhance this algorithm by adding a key, making it a key-based encryption algorithm, thus obtaining a new encryption algorithm

Keywords: Digital image, permutation, chaotic map

INTRODUCTION

In the digital era, the widespread use of images for communication, data representation, and information storage has necessitated the development of robust techniques to secure the content within these visual assets. As the significance of digital images continues to grow across various domains, concerns related to privacy, authentication, and intellectual property rights have intensified. In response to these challenges, researchers have explored a myriad of image-hiding and encryption methods to safeguard sensitive information embedded within images. One intriguing approach that has gained attention is the manipulation of pixel locations through permutation, offering a novel perspective on image feature hiding. Pixel permutation involves the rearrangement of pixel positions within an image, creating a visually altered representation while preserving the essential features. This technique goes beyond traditional encryption methods and introduces a layer of complexity that enhances the concealment of critical information, making it more resistant to unauthorized access. The objective of this research is to delve into the theoretical foundations and practical applications of image feature hiding through the permutation of pixel locations. By investigating the inherent properties of pixel rearrangement, we aim to uncover its potential as a robust and versatile method for concealing information within digital images. Furthermore, we will explore the impact of pixel permutation on key image attributes, such as visual quality, computational efficiency, and resistance to various attacks. This research will contribute to the growing body of knowledge in image security, providing insights into the effectiveness of pixel permutation as a means of hiding sensitive features. As the digital landscape continues to evolve, understanding and refining innovative techniques like pixel location permutation will play a crucial role in addressing emerging challenges associated with image privacy and data protection.

With the growing utilization of digital images across various domains, safeguarding their security has become a paramount concern. The advent of the big data era has elevated the significance of digital images as pivotal information carriers, making their security crucial in thwarting unauthorized access, misuse, and breaches. The transmission and storage of digital images are susceptible to technical glitches or illicit user attacks, posing severe consequences. To tackle these challenges, we propose an innovative image encryption algorithm that integrates chaotic image encryption with a convolutional neural network (CNN). The suggested algorithm employs chaotic mapping to generate a random series, subsequently used for encrypting the original image, ensuring a high degree of randomness and unpredictability [1]. A CNN is leveraged to bolster the security and efficiency of the encryption algorithm by autonomously extracting image features. The algorithm segments the original image into small blocks for local encryption, executes pixel replacement and dissimilarity operations using the random series from chaotic mapping, and feeds the encrypted chunks into the neural network for convolution to extract high-level image features. Finally, the encrypted chunks are amalgamated to produce an encrypted image. The primary advantages of the proposed algorithm encompass heightened randomness and unpredictability, courtesy of the chaotic mapping-generated random series. It also exhibits enhanced efficiency and accuracy through the CNN's deep learning of the image. Additionally, the algorithm is characterized by its simplicity, reliability, and adaptability to diverse image encryption scenarios. The research outcomes of the algorithm can be extrapolated to areas beyond image encryption, such as video encryption and audio encryption, offering substantial application prospects, [2] [3] [4] [5].

Cloud computing has grown to be more and more popular in recent years because of the speedy improvement and maturation of cloud garage services, including Baidu Cloud, Amazon Simple Storage Service, Windows Azure, and Google Cloud. Cloud storage saves consumer facts on a cloud server, and the company performs operations on the statistics through an online community environment, charging the person for hardware sources and service time. Cloud storage is widely used due to its advantages, such as scalability, accessibility, shareability, and constant backup of massive amounts of information. However, there are nevertheless significant problems and demanding situations related to cloud storage, along with the vulnerability of transmitted facts to intrusion and the lack of control over the facts. To deal with those worries, an honest method of encrypting personal data before outsourcing it to the cloud has been proposed. However, attempting to find encrypted facts inside the cloud remains a hard task. Goldreich first proposed the ciphertext seek mechanism in 1996; however, it required many interactions between the customer and server, making it impractical. Song supplied a realistic searchable encryption generation in 2000, which has become a milestone in the improvement of searchable encryption. Searchable encryption technology includes two types: symmetric searchable encryption (SSE) and public key encryption with keyword seek (PEKS). PEKS has a broader application prospect than SSE; however, it also has the trouble of securely sharing keys for information encryption. Although there have been many survey studies on searchable encryption, there are few entire surveys on PEKS. This paper aims to supplement these surveys by providing a complete look at PEKS schemes [6] [7] [8] [9].

Reversible statistics hiding (RDH) is a technique that allows embedding mystery facts in a cover signal, such as an image, while maintaining the authentic signal's best after ideal extraction of the hidden statistics. Reversible records hiding in encrypted pictures (RDHEI) is a combination of RDH and encryption in which the name of the game information is embedded in the encrypted photograph. RDHEI has numerous packages, which include copyright safety, authentication, cover communication, and more. Several approaches have been proposed for RDHEI, which include: 1. Reserving room before encryption (RRBE): This method vacates room inside the encrypted image before encryption, which can be used to embed mystery statistics. 2. Vacating room after encryption (VRAE): This approach vacates room in the encrypted photograph after encryption, which can be used to embed mystery statistics. 3. Compressing the encrypted image: This technique compresses the encrypted photograph, which can be used to vacate rooms and embed mystery data. 4. Exploiting the correlation between neighboring pixels: This technique exploits the correlation between neighboring pixels within the encrypted photo to embed mystery records. RDHEI schemes can be categorized into separable or joint techniques. Separable techniques perform statistics extraction and photo decryption one after the other, while joint techniques perform them together. Several RDHEI schemes have been proposed, including: 1. These schemes use VRAE and exploit the correlation among neighboring pixels to embed secret records. They also use disbursed supply coding to enhance the hiding ability. 2. This scheme uses the correlation of pattern pixels and non-pattern ones to embed mystery records. It also uses a move cipher to encrypt sample pixels and a selected encryption technique to encrypt prediction errors of non-pattern ones. 3. This scheme uses a similar random integer to encrypt complete pixels in a

block, which preserves the correlation of pixels in a block and permits embedding secret statistics. 4. This scheme makes use of block permutation to shield against a designated plaintext attack. 5. This scheme makes use of a changed model of histogram transferring and the distinction growth method to embed mystery records within the encrypted picture. 6. This scheme improves two preceding schemes in terms of the hiding capability and accuracy of information extraction and authentic photo recovery. It exploits several binary public keys to embed information in line with the criterion of maximizing the minimal Hamming distance among all keys. 7. This scheme proposes an excessive embedding capability scheme that exploits the correlation of pixels in a block to embed mystery statistics in the encrypted image. In summary, RDHEI is a method that allows embedding mystery statistics in an encrypted photograph while preserving the authentic image's first-rate. Various strategies were proposed for RDHEI, such as RRBE, VRAE, compressing the encrypted image, and exploiting the correlation between neighboring pixels. These schemes can be categorized into separable or joint techniques, and they have numerous packages in copyright protection, authentication, cover communication, and more [10] [11] [12].

The development of the information and network era has greatly facilitated the fast evolution of the Internet, serving as the technical spine deeply ingrained in all facets of human existence. This evolution has led to the generation of numerous data bureaucracies and enormous volumes of statistics on a daily basis. Given the close affiliation of these statistics with user statistics, safeguarding them will become specifically critical. Digital picture records, being an enormous data provider, hold a good function in community transmission. Encrypting images stands proudly as a vital technique to ensure the security of these photos. Image facts possess characteristics such as strong pixel correlation, excessive record redundancy, and large volumes of records. Traditional textual content encryption algorithms like DES and AES are unsuitable for image encryption because of these specific functions. Over the past few years, the mainstream of cryptography has shifted towards photo encryption techniques primarily based on chaotic systems, mobile automata, DNA encoding, bit-plane decomposition, and elliptic curves. Chaotic systems, chosen for their unpredictability, ergodicity, and preliminary country sensitivity, have grown to be a favored preference for encryption. However, in maximum chaotic photo encryptions, the chaotic sequence is transformed into a chunk collection for plaintext encryption, and the safety of the encryption relies on the residences of this bit series [13]. The vulnerability of chaotic cryptosystems to equivalent keys arises from the independence of the encryption system from plaintext and/or ciphertext. Additionally, elliptic curve cryptography, despite providing excessive safety, is extra complex and prone to mistakes as compared to other cryptosystems with the same key size, thereby compromising its security. Since Matthews added the generalized logistic map for generating pseudorandom numbers in record encryption, several scholars have engaged in chaotic image encryption schemes. Studies have reviewed the principle and alertness of chaotic cryptography, focusing on high-dimensional chaotic cryptography's progress and its application in stable multimedia verbal exchange and hardware implementation technology. Researchers have stated that various chaotic photo encryption algorithms evolved within the past many years, categorizing and summarizing the generally used chaotic systems, diffusion operations, and evaluation techniques in element [14]. Generally, chaotic encryption algorithms employing multi-spherical permutation-diffusion techniques provide advanced cryptographic residences compared to those with a single round, making them resilient against chosen-plaintext assaults. Cryptography and cryptanalysis are intertwined, each influencing and advancing the other. Cryptanalysis, adhering to Kerchhoff's precept, has ambitions for a cryptographic gadget's security even if the entirety of the device, besides the important thing, is publicly acknowledged. Different assaults, together with ciphertext-handiest, regarded-plaintext, selected-plaintext, and chosen-ciphertext assaults, are essential in cryptology. Recent years have seen the emergence of linear and differential assaults, significantly impacting cryptanalysis. Many new analysis methods are derivatives of those procedures [15] [16] [17].

Due to the rapid surge in information transmission through available insecure channels, it has become crucial to safeguard information confidentiality against arbitrary intrusions that could compromise user privacy. In our daily lives, digital images play a significant role in communication networks, making it imperative to ensure the security of transmitted information on open networks, such as the internet. Information protection stands out as the optimal solution to meet these security requirements. Protecting digital images can be approached in two ways: image hiding, encompassing techniques like watermarking and steganography, and image encryption. The primary distinction lies in the focus of image hiding, which aims to keep the existence of an image secret, while encryption safeguards the content of an image without concealing its existence. Image encryption commonly employs stream ciphers based on pseudorandom bit sequences or keys. Two types of keys are utilized for encrypting image data: symmetric keys and asymmetric keys. Symmetric keys involve using the same keys for encryption and decryption,

while asymmetric keys differ between the two processes. Stream ciphers, a type of symmetric cipher, encrypt one bit at a time. As the name suggests, stream ciphers utilize a stream of key bits generated by a key stream generator. Secret key-based stream ciphers are often created using Linear Feedback Shift Registers (LFSR). Enhancing the security of stream ciphers involves the use of nonlinear keys, achievable through the combination of multiple LFSRs. This paper proposes an encryption algorithm to achieve secure image encryption, employing multi-level block permutation and a nonlinear key stream cipher. The secret keys are generated using a nonlinear filter generator based on LFSRs, each corresponding to the size of digital image pixels [18], [19].

METHODOLOGY

We will assume the following hypothetical scenario which is represented by a digital matrix we perform the five permutation stages on the pixels these stages are reversible allowing the original image to be recovered before the permutations after that in the results section we will see the application of this algorithm to five images.

Methodology of proposed algorithm of permutation of pixels locations:

1.1. The Steps of Algorithm in general:

The main steps in general that are used in our algorithm to encrypt the image:-

Step-1:- Replace the first column with the last column, the third column with the third column from the last, and so on by contrasting and alternating by taking one column and leaving the next.

Step-2:- Replace the first Row with the last Row, the third Row with the third Row from the last, and so on by contrasting and alternating by taking one Row and leaving the next.

Step-3:- Swap the second column with the second column after the middle, then the fourth column with the fourth column after the middle, and so on. We swap one column and leave one column until we reach the middle.

Step-4:- Swap the second Row with the second Row after the middle, then the fourth Row with the fourth Row after the middle, and so on. We swap one Row and leave one Row until we reach the middle.

Step-5:- Performing a transformation using the logistic function, which is the transformation that has the most impact on the image's features. It is a switch that works on the three layers of the image, each layer independently. Which leads to great encryption of image features by 100%.

1.2. Practical example

We assume that the matrix of color values for the original image to be concealed through substitutions is

$$\text{Let the matrix of origion image } A_{(6,9)} = \begin{bmatrix} 2 & 3 & 5 & 5 & 2 & 3 & 2 & 1 & 5 \\ 4 & 6 & 8 & 1 & 9 & 7 & 8 & 9 & 4 \\ 9 & 7 & 4 & 8 & 2 & 1 & 4 & 5 & 2 \\ 8 & 8 & 9 & 8 & 7 & 7 & 8 & 5 & 5 \\ 4 & 5 & 6 & 4 & 5 & 6 & 2 & 7 & 8 \\ 2 & 1 & 3 & 3 & 5 & 4 & 9 & 4 & 6 \end{bmatrix}$$

1.2.1. First: Hiding by permutation of pixels locations

The first step in permutations: is it involves interchanging rows where the first row will be replaced by the sixth row the sixth row will be replaced by the first row the third row will be replaced by the fourth row and the fourth row will be replaced by the third row

$$\text{First permutations } B1_{(6,9)} = \begin{bmatrix} 2 & 1 & 3 & 3 & 5 & 4 & 9 & 4 & 6 \\ 4 & 6 & 8 & 1 & 9 & 7 & 8 & 9 & 4 \\ 8 & 8 & 9 & 8 & 7 & 7 & 8 & 5 & 5 \\ 9 & 7 & 4 & 8 & 2 & 1 & 4 & 5 & 2 \\ 4 & 5 & 6 & 4 & 5 & 6 & 2 & 7 & 8 \\ 2 & 3 & 5 & 5 & 2 & 3 & 2 & 1 & 5 \end{bmatrix}$$

The second step in the permutations: is to swap between the columns. The first column will be replaced by the ninth column, and the ninth column will be replaced by the first column. Similarly, the third column will be replaced by the seventh column, and the seventh column will be replaced by the third column.

$$\text{Second permutations } B2_{(6,9)} = \begin{bmatrix} 6 & 1 & 9 & 3 & 5 & 4 & 3 & 4 & 2 \\ 4 & 6 & 8 & 1 & 9 & 7 & 8 & 9 & 4 \\ 5 & 8 & 8 & 8 & 7 & 7 & 9 & 5 & 8 \\ 2 & 7 & 4 & 8 & 2 & 1 & 4 & 5 & 9 \\ 8 & 5 & 2 & 4 & 5 & 6 & 6 & 7 & 4 \\ 5 & 3 & 2 & 5 & 2 & 3 & 5 & 1 & 2 \end{bmatrix}$$

The third step in the permutations: is to swap between the rows for a second time in a new pattern. The second row will be replaced by the fifth row, and the fifth row will be replaced by the second row. This pattern continues if the matrix is larger by skipping rows and swapping them while maintaining the same pace between the upper and lower halves.

$$\text{Third permutations } B3_{(6,9)} = \begin{bmatrix} 6 & 1 & 9 & 3 & 5 & 4 & 3 & 4 & 2 \\ 8 & 5 & 2 & 4 & 5 & 6 & 6 & 7 & 4 \\ 5 & 8 & 8 & 8 & 7 & 7 & 9 & 5 & 8 \\ 2 & 7 & 4 & 8 & 2 & 1 & 4 & 5 & 9 \\ 4 & 6 & 8 & 1 & 9 & 7 & 8 & 9 & 4 \\ 5 & 3 & 2 & 5 & 2 & 3 & 5 & 1 & 2 \end{bmatrix}$$

The fourth step in the permutations: is to swap between the columns for a second time in a new pattern. The second column will be replaced by the sixth column, and the sixth column will be replaced by the second column. Similarly, the fourth column will be replaced by the eighth column, and the eighth column will be replaced by the fourth column. This pattern continues if the matrix is larger by skipping columns and swapping them while maintaining the same pace between the left and right halves.

$$\text{Fourth permutations } B4_{(6,9)} = \begin{bmatrix} 6 & 4 & 9 & 4 & 5 & 1 & 3 & 3 & 2 \\ 8 & 6 & 2 & 7 & 5 & 5 & 6 & 4 & 4 \\ 5 & 7 & 8 & 5 & 7 & 8 & 9 & 8 & 8 \\ 2 & 1 & 4 & 5 & 2 & 7 & 4 & 8 & 9 \\ 4 & 7 & 8 & 9 & 9 & 6 & 8 & 1 & 4 \\ 5 & 3 & 2 & 1 & 2 & 3 & 5 & 5 & 2 \end{bmatrix}$$

The fifth step in the permutations: involves using a chaotic map. This function performs a chaotic permutation on the pixels, completely hiding all image features in the case of the last matrix, $b = 4 \ 6 \ 9$. The resulting image, under the influence of the chaotic map, will be as follows:

$$\text{Fifth permutations by chaotic map } B4_{(6,9)} = \begin{bmatrix} 2 & 9 & 3 & 2 & 8 & 2 & 7 & 5 & 9 \\ 5 & 4 & 8 & 2 & 1 & 2 & 3 & 1 & 8 \\ 8 & 4 & 9 & 1 & 4 & 6 & 4 & 3 & 6 \\ 5 & 2 & 7 & 8 & 5 & 4 & 9 & 1 & 8 \\ 5 & 6 & 4 & 3 & 8 & 7 & 5 & 8 & 5 \\ 4 & 7 & 9 & 6 & 7 & 5 & 2 & 4 & 5 \end{bmatrix}$$

The last matrix represents the latest version of the original matrix, and below is the comparison between the original matrix and the last matrix after performing five swap operations:

Table 1: Comparison between the original matrix and the matrix after five swap steps.

The origon matrix	The final matrix after five permutations
$\begin{bmatrix} 2 & 3 & 5 & 5 & 2 & 3 & 2 & 1 & 5 \\ 4 & 6 & 8 & 1 & 9 & 7 & 8 & 9 & 4 \\ 9 & 7 & 4 & 8 & 2 & 1 & 4 & 5 & 2 \\ 8 & 8 & 9 & 8 & 7 & 7 & 8 & 5 & 5 \\ 4 & 5 & 6 & 4 & 5 & 6 & 2 & 7 & 8 \\ 2 & 1 & 3 & 3 & 5 & 4 & 9 & 4 & 6 \end{bmatrix}$	$\begin{bmatrix} 2 & 9 & 3 & 2 & 8 & 2 & 7 & 5 & 9 \\ 5 & 4 & 8 & 2 & 1 & 2 & 3 & 1 & 8 \\ 8 & 4 & 9 & 1 & 4 & 6 & 4 & 3 & 6 \\ 5 & 2 & 7 & 8 & 5 & 4 & 9 & 1 & 8 \\ 5 & 6 & 4 & 3 & 8 & 7 & 5 & 8 & 5 \\ 4 & 7 & 9 & 6 & 7 & 5 & 2 & 4 & 5 \end{bmatrix}$

1.2.2. Second: Reverse permutations

The original matrix can be restored by reversing each swap step that we performed above from the last step to the first step sequentially. The results will be as follows:








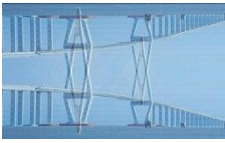


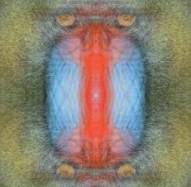

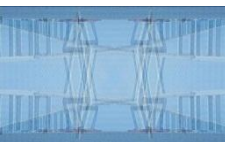














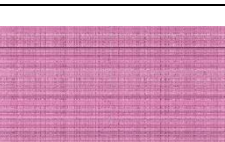
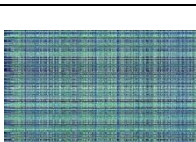
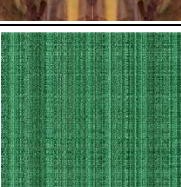
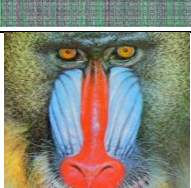

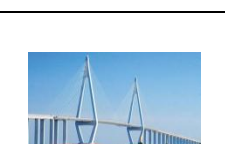


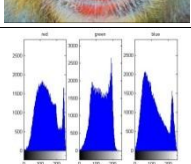
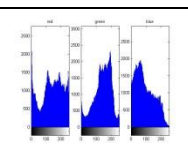
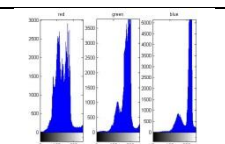
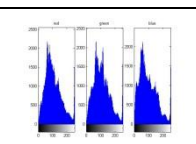
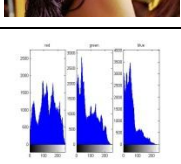
Reverse the permutation steps from the last to the first in succession	The resulting matrix after reversing the permutations
<i>Fifth permutations by chaotic map $B4_{(6,9)}$</i>	$\begin{bmatrix} 2 & 9 & 3 & 2 & 8 & 2 & 7 & 5 & 9 \\ 5 & 4 & 8 & 2 & 1 & 2 & 3 & 1 & 8 \\ 8 & 4 & 9 & 1 & 4 & 6 & 4 & 3 & 6 \\ 5 & 2 & 7 & 8 & 5 & 4 & 9 & 1 & 8 \\ 5 & 6 & 4 & 3 & 8 & 7 & 5 & 8 & 5 \\ 4 & 7 & 9 & 6 & 7 & 5 & 2 & 4 & 5 \end{bmatrix}$
<i>Fourth permutations $B4_{(6,9)}$</i>	$\begin{bmatrix} 6 & 4 & 9 & 4 & 5 & 1 & 3 & 3 & 2 \\ 8 & 6 & 2 & 7 & 5 & 5 & 6 & 4 & 4 \\ 5 & 7 & 8 & 5 & 7 & 8 & 9 & 8 & 8 \\ 2 & 1 & 4 & 5 & 2 & 7 & 4 & 8 & 9 \\ 4 & 7 & 8 & 9 & 9 & 6 & 8 & 1 & 4 \\ 5 & 3 & 2 & 1 & 2 & 3 & 5 & 5 & 2 \end{bmatrix}$
<i>Third permutations $B3_{(6,9)}$</i>	$\begin{bmatrix} 6 & 1 & 9 & 3 & 5 & 4 & 3 & 4 & 2 \\ 8 & 5 & 2 & 4 & 5 & 6 & 6 & 7 & 4 \\ 5 & 8 & 8 & 8 & 7 & 7 & 9 & 5 & 8 \\ 2 & 7 & 4 & 8 & 2 & 1 & 4 & 5 & 9 \\ 4 & 6 & 8 & 1 & 9 & 7 & 8 & 9 & 4 \\ 5 & 3 & 2 & 5 & 2 & 3 & 5 & 1 & 2 \end{bmatrix}$
<i>Second permutations $B2_{(6,9)}$</i>	$\begin{bmatrix} 6 & 1 & 9 & 3 & 5 & 4 & 3 & 4 & 2 \\ 4 & 6 & 8 & 1 & 9 & 7 & 8 & 9 & 4 \\ 5 & 8 & 8 & 8 & 7 & 7 & 9 & 5 & 8 \\ 2 & 7 & 4 & 8 & 2 & 1 & 4 & 5 & 9 \\ 8 & 5 & 2 & 4 & 5 & 6 & 6 & 7 & 4 \\ 5 & 3 & 2 & 5 & 2 & 3 & 5 & 1 & 2 \end{bmatrix}$
<i>First permutations $B1_{(6,9)}$</i>	$\begin{bmatrix} 2 & 1 & 3 & 3 & 5 & 4 & 9 & 4 & 6 \\ 4 & 6 & 8 & 1 & 9 & 7 & 8 & 9 & 4 \\ 8 & 8 & 9 & 8 & 7 & 7 & 8 & 5 & 5 \\ 9 & 7 & 4 & 8 & 2 & 1 & 4 & 5 & 2 \\ 4 & 5 & 6 & 4 & 5 & 6 & 2 & 7 & 8 \\ 2 & 3 & 5 & 5 & 2 & 3 & 2 & 1 & 5 \end{bmatrix}$
The matrix after reversing all the permutation steps It matches the original matrix with no error	$\begin{bmatrix} 2 & 3 & 5 & 5 & 2 & 3 & 2 & 1 & 5 \\ 4 & 6 & 8 & 1 & 9 & 7 & 8 & 9 & 4 \\ 9 & 7 & 4 & 8 & 2 & 1 & 4 & 5 & 2 \\ 8 & 8 & 9 & 8 & 7 & 7 & 8 & 5 & 5 \\ 4 & 5 & 6 & 4 & 5 & 6 & 2 & 7 & 8 \\ 2 & 1 & 3 & 3 & 5 & 4 & 9 & 4 & 6 \end{bmatrix}$

This result indicates the quality and accuracy of the algorithm, as well as its error-free nature.

EXPERIMENTAL RESULTS AND ANALYSIS

This algorithm was implemented using MATLAB software[25], and we applied the process to five images: Suspension Bridge, Floating Bridge, Child, Lena, and Baboon. To further clarify the workflow steps, we provide below the complete results for each stage of permutations.

Table 2: Comprehensive breakdown of algorithm implementation on five images (Baboon, Child, Suspension Bridge, Floating Bridge, Lena) including all its histogram.

Image Name	Baboon	Child	Suspension bridge	floating bridge	Lena
Origin Image					
First permutation					
Second permutation					
Third permutation					
Fourth permutation					
Fifth permutation by chaotic map					
image after depermutation					
Histogram is the same for all the images above					

In the first second third and fourth permutations we flip the pixels in the three color channels of the image simultaneously resulting in the same colors as the original image but in a chaotic arrangement this is because the combinations of the three primary colors red green and blue in each pixel remain unchanged this makes these permutations weak and unsuitable as encryption algorithms when linked to an encryption key however in the fifth

permutation we scatter the positions of the pixels in each color channel independently using logistic equations and chaotic maps this leads to a change in the combinations of the three primary colors in each pixel resulting in very sophisticated results where the original image features completely disappear and the new image does not indicate the original image at all this strength and quality make our algorithm along with the fifth permutation a valid encryption algorithm after adding an encryption key regarding the histogram we observe that it matches for all images the original image the five permutations and the image after reversing the permutations this is because our process involved scattering the positions of the pixels or the color components of the pixels without changing the color intensity in those pixels In the second table below, we will see the time taken for each permutation, the time for the final reversal of permutations, and the time for recovering the original image. Further, through the subsequent tables, we will examine all the readings of international accuracy standards for the purpose of comparisons or as a basis for future research conducted by researchers.

Table 3: on five images (Baboon, Child, Suspension Bridge, Floating Bridge, Lena) including all data and resulting readings from the algorithm execution.

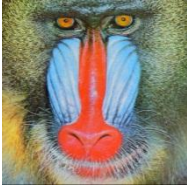









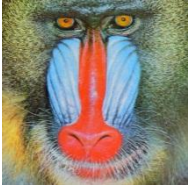




Image Name	Baboon 	Child 	Suspension bridge 	floating bridge 	Lena 
First permutation time	0.0740	0.0540	0.0660	0.0420	0.0630
Second permutation time	0.0920	0.0720	0.0850	0.0580	0.0810
Third permutation time	0.1380	0.1150	0.1200	0.0930	0.1260
Fourth permutation time	0.1580	0.1390	0.1370	0.1100	0.1450
Fifth permutation time	0.1820	0.1640	0.1770	0.1520	0.1720
Time to reverse all permutations	0.1790	0.1730	0.1820	0.1490	0.1750

Table 4: Mean error and MSE that resulting readings from the algorithm execution. on five images (Baboon, Child, Suspension Bridge, Floating Bridge, Lena)

Image Name	Baboon 	Child 	Suspension bridge 	floating bridge 	Lena 
Mean error between origin and permutation1	26.7831	30.5987	17.6900	29.7886	26.8956
Mean error between origin and permutation2	38.3347	46.2183	24.7068	47.5125	46.6513
Mean error between origin and permutation3	56.7229	66.9638	31.1709	54.5805	53.3909

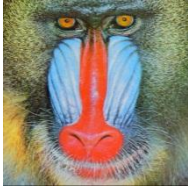




Mean error between origin and permutation4	61.0937	72.9342	36.0128	56.1843	58.9604
Mean error between origin and permutation5	200.9874	202.3658	201.2314	199.3254	212.5411
Mean error between origin and after depermutation	0	0	0	0	0
MSE between origin and permutation1	2.3767e+03	3.0191e+03	1.1856e+03	2.9202e+03	2.4209e+03
MSE between origin and permutation2	3.2995e+03	4.4770e+03	1.6160e+03	4.6974e+03	4.7569e+03
MSE between origin and permutation3	5.2158e+03	8.0164e+03	2.0684e+03	5.4246e+03	5.3465e+03
MSE between origin and permutation4	5.8201e+03	8.9245e+03	2.4310e+03	5.4553e+03	5.9570e+03
MSE between origin and permutation5	4.0395e+4	4.0951e+4	4.0494e+4	3.9730e+4	4.5173e+4
MSE between origin and after depermutation	0	0	0	0	0

Table 5: PSNR, Entropy, and Standard deviation that resulting readings from the algorithm execution. on five images (Baboon, Child, Suspension Bridge, Floating Bridge, Lena)

Image Name	Baboon 	Child 	Suspension bridge 	floating bridge 	Lena 
PSNR between origin and permutation1	31.2510	30.7314	32.7612	30.8038	31.2109
PSNR between origin and permutation2	30.5386	29.8759	32.0886	29.7715	29.7442
PSNR between origin and permutation3	29.5442	28.6109	31.5526	29.4590	29.4905
PSNR between origin and permutation4	29.3062	28.3779	31.2018	29.4467	29.2557

PSNR between origin and permutation5	7.0215	6.9549	8.5521	4.1257	5.9852
PSNR between origin and after depermutation	∞	∞	∞	∞	∞
Entropy of origin image	0.0303	0.0748	0.0280	0.0151	0.1381
Entropy of permutation1	0.0303	0.0748	0.0280	0.0151	0.1381
Entropy of permutation2	0.0303	0.0748	0.0280	0.0151	0.1381
Entropy of permutation3	0.0303	0.0748	0.0280	0.0151	0.1381
Entropy of permutation4	0.0303	0.0748	0.0280	0.0151	0.1381
Entropy of permutation5	0.0303	0.0748	0.0280	0.0151	0.1381
Entropy of image after depermutation	0.0303	0.0748	0.0280	0.0151	0.1381
Standard deviation of origin image	55.9074	67.7219	47.3612	55.3216	63.7814
Standard deviation of permutation1	55.9074	67.7219	47.3612	55.3216	63.7814
Standard deviation of permutation2	55.9074	67.7219	47.3612	55.3216	63.7814
Standard deviation of permutation3	55.9074	67.7219	47.3612	55.3216	63.7814
Standard deviation of permutation4	55.9074	67.7219	47.3612	55.3216	63.7814
Standard deviation of permutation5	55.9074	67.7219	47.3612	55.3216	63.7814
Standard deviation of image after depermutation	55.9074	67.7219	47.3612	55.3216	63.7814

Table 6: Correlation coefficient , NPCR, UACI, and Dimension of image that resulting readings from the algorithm execution. on five images (Baboon, Child, Suspension Bridge, Floating Bridge, Lena)

Image Name	Baboon 	Child 	Suspension bridge 	floating bridge 	Lena 
Correlation coefficient between origin and permutation1	0.6198	0.6709	0.7357	0.5229	0.7025
Correlation coefficient between origin and permutation2	0.4722	0.5119	0.6398	0.2326	0.4153
Correlation coefficient between origin and permutation3	0.1656	0.1260	0.5389	0.1138	0.3429
Correlation coefficient between origin and permutation4	0.0690	0.0270	0.4581	0.1088	0.2678
Correlation coefficient between origin and permutation5	0.0025	0.0029	0.0031	0.0010	0.0012
Correlation coefficient between origin and after depermutation	1	1	1	1	1
NPCR between origin and permutation1	49.6014	49.4566	49.1783	49.9078	49.4914
NPCR between origin and permutation2	74.3785	74.4975	73.8192	74.6501	74.4893
NPCR between origin and permutation3	99.2479	86.9103	86.2724	87.0136	86.7925

NPCR between origin and permutation4	99.3927	93.2277	98.6550	93.1473	93.1353
NPCR between origin and permutation5	99.9987	99.9898	99.9198	99.8219	99.7899
NPCR between origin and after depermutation	0	0	0	0	0
Dimension of image	511 513 3	446 598 3	361 599 3	358 597 3	509 510 3

CONCLUSIONS

From the above tables, we can deduce numerous important results that we will discuss in detail as follows:

- Table 2 illustrates the progression of the image scattering process during the ongoing iterations on pixel locations, especially in the fifth iteration, which completely hides all image features. This qualifies this algorithm to be an excellent encryption algorithm when combined with a creative encryption key. According to the subsequent researcher's opinion, who may rely on this research, this algorithm serves as a gateway to dozens, if not hundreds, of research studies in the field of hiding and encryption.
- Similarly, to Table 2 and the histogram matching for all images. original image after the first iteration image after the second iteration image after the third iteration image after the fourth iteration image after the fifth iteration and the image after reversing all five iterations we noticed that the matching indicates the intensity of each color in each pixel that has not been manipulated and remains unchanged however what occurred is the scattering of the positions of each color intensity for each pixel and for each band independently this result indicates the possibility of encryption without the need to change the color values of the pixels as is commonly done in most algorithms.
- We observe from table 3 that the time for each iteration for all images is close and very minimal ranging from 0.14 to 0.18 this indicates that this algorithm is practical and excellent qualifying it to be a reliable and excellent encryption algorithm when combined with an encryption key.
- In table 4, it is evident that the mean error and consequently the mean squared error MSE increase as we progress through the iterations from the first iteration to the fifth iteration it reaches its peak in the fifth iteration which is considered the best and most efficient among all preceding iterations this result indicates that the efficiency of this iteration algorithm increases step by step reaching the most efficient stage in the fifth step.
- From table 4, we observe that the mean error and MSE between the original image and the image after reversing all the swaps and recovering the image are both zero. This indicates that there is no loss of information during the swapping and reversing operations and that the color values in each pixel have returned without any change and are identical between the original image and the image after reversing the swaps. This may make the algorithm a reliable candidate for encryption algorithms.
- In table 5, we notice that the PSNR falls within the natural range between the original image and the image for each swap from the first to the fifth. The reason for this reading is that there is no change in the brightness intensities of the color values of the pixels but only a change in their positions. Similarly, this is related to the value of the mean error mentioned in table 3, where we see it is close in the first, second, third, and fourth swaps with some decrease after each swap, but we see its value decreases significantly in the fifth swap, and the reason is the very low reading of the mean error for the fifth swap.

- Also, in table 5, we notice that the PSNR equals infinity between the original image and the image after reversing all the swaps due to the absence of any loss of information during the swaps and reversals, resulting in a mean error value of zero.
- Furthermore, in table 5, we observe that the entropy and standard deviation for each level of swapping the original image the first swap image, the second swap image the third swap image the fourth swap image The fifth swap image and the image after reversing all the swaps are equal. This is because these two measures are not affected or changed by the changes in the color values and pixel positions as long as the brightness intensities are constant and only their positions change during each stage of swapping. These two measures do not change their readings. It is worth mentioning that these two measures are sensitive to any change; if there is a slight change in the brightness intensities within the colors of some pixels, it will be reflected in their results.
- In table number 6, we observe that the reading of the correlation coefficient, which represents the magnitude of the relationship between two images, decreases as the number of substitution steps increases from the first substitution to the fifth substitution. The lowest value is found in the fifth substitution, indicating a weak relationship between the original image and the image after the fifth substitution.
- Similarly, in table number 6, as evidence of no loss of information during the substitution process, we notice that the reading of the correlation coefficient between the original image and the image after reversing all substitutions is at its maximum value of 1. This is the highest value for this coefficient, indicating a match between the two images, the original image and the image after reversing all substitutions.
- Furthermore, in Table 6, we observe that the value of the NPCR number of pixel change rate increases as we progress in the substitutions, reaching its highest value of 99 in the fifth substitution. This coefficient is used to test the strength of encryption algorithms. If the value of this coefficient between the original image and the image after encryption is 99, then the encryption algorithm is considered strong in the case of a match between the compared images. The value of the NPCR is zero, as is the case between the original image and the image after reversing all substitutions, as shown in Table 6.

REFERENCES

- [1] L. Feng, J. Du, C. Fu, and W. Song, "Image Encryption Algorithm Combining Chaotic Image Encryption and Convolutional Neural Network," *Electronics (Switzerland)*, vol. 12, no. 16, Aug. 2023, doi: 10.3390/electronics12163455.
- [2] H. Qiu, X. Xu, Z. Jiang, K. Sun, and C. Cao, "Dynamical behaviors, circuit design, and synchronization of a novel symmetric chaotic system with coexisting attractors," *Sci Rep*, vol. 13, no. 1, Dec. 2023, doi: 10.1038/s41598-023-28509-z.
- [3] O. Ozmen Garibay et al., "Six Human-Centered Artificial Intelligence Grand Challenges," *Int J Hum Comput Interact*, vol. 39, no. 3, pp. 391–437, 2023, doi: 10.1080/10447318.2022.2153320.
- [4] M. Al-Zubaidie, Z. Zhang, and J. Zhang, "Efficient and Secure ECDSA Algorithm and its Applications: A Survey," 2019.
- [5] Y. Hong, Y. Wang, J. Su, Y. Wen, and Z. Yang, "Image Encryption Algorithm Based on Chaotic Mapping and Binary Bidirectional Zigzag Transform," *IEEE Access*, vol. 11, pp. 78498–78510, 2023, doi: 10.1109/ACCESS.2023.3299503.
- [6] Y. Zhou, N. Li, Y. Tian, D. An, and L. Wang, "Public key encryption with keyword search in cloud: A survey," *Entropy*, vol. 22, no. 4, Apr. 2020, doi: 10.3390/E22040421.
- [7] Y. Lu, G. Wang, and J. Li, "Keyword guessing attacks on a public key encryption with keyword search scheme without random oracle and its improvement," *Inf Sci (N Y)*, vol. 479, pp. 270–276, Apr. 2019, doi: 10.1016/j.ins.2018.12.004.
- [8] T. A. Al-Maadeed, I. Hussain, A. Anees, and M. T. Mustafa, "A image encryption algorithm based on chaotic Lorenz system and novel primitive polynomial S-boxes," *Multimed Tools Appl*, vol. 80, no. 16, pp. 24801–24822, Jul. 2021, doi: 10.1007/s11042-021-10695-5.
- [9] A. Mansouri and X. Wang, "Image encryption using shuffled Arnold map and multiple values manipulations," *Visual Computer*, vol. 37, no. 1, pp. 189–200, Jan. 2021, doi: 10.1007/s00371-020-01791-y.

-
- [10] A. Mohammadi and M. Nakhkash, "Reversible Data Hiding in Encrypted Images using Local Difference of Neighboring Pixels," arXiv preprint arXiv:1907.05123. 2019 Jul 11, 2019.
 - [11] Y. Hu, H. Wu, and L. Zhou, "A Novel Hyperchaotic 2D-SFCF with Simple Structure and Its Application in Image Encryption," *Entropy*, vol. 24, no. 9, Sep. 2022, doi: 10.3390/e24091266.
 - [12] X. Cao, L. Du, X. Wei, D. Meng, and X. Guo, "High Capacity Reversible Data Hiding in Encrypted Images by Patch-Level Sparse Representation," *IEEE Trans Cybern*, vol. 46, no. 5, pp. 1132–1143, May 2016, doi: 10.1109/TCYB.2015.2423678.
 - [13] G. Shi, S. Yu, and Q. Wang, "Security Analysis of the Image Encryption Algorithm Based on a Two-Dimensional Infinite Collapse Map," *Entropy*, vol. 24, no. 8, Aug. 2022, doi: 10.3390/e24081023.
 - [14] J. Oravec, L. Ovsenik, and J. Papaj, "An image encryption algorithm using logistic map with plaintext-related parameter values," *Entropy*, vol. 23, no. 11, Nov. 2021, doi: 10.3390/e23111373.
 - [15] T. Li and D. Zhang, "Hyperchaotic image encryption based on multiple bit permutation and diffusion," *Entropy*, vol. 23, no. 5, May 2021, doi: 10.3390/e23050510.
 - [16] S. Roy, M. Shrivastava, U. Rawat, C. V. Pandey, and S. K. Nayak, "IESCA: An efficient image encryption scheme using 2-D cellular automata," *Journal of Information Security and Applications*, vol. 61, Sep. 2021, doi: 10.1016/j.jisa.2021.102919.
 - [17] J. Xu, B. Zhao, and Z. Wu, "Research on Color Image Encryption Algorithm Based on Bit-Plane and Chen Chaotic System," *Entropy*, vol. 24, no. 2, Feb. 2022, doi: 10.3390/e24020186.
 - [18] A. Jasim and H. Hakim, "Multilevel Permutation with Different Block Size/ Stream Cipher Image Encryption," *Iraqi Journal for Electrical and Electronic Engineering*, vol. 11, no. 1, pp. 42–48, Jun. 2015, doi: 10.37917/ijeee.11.1.5.
 - [19] N. A. Azam, U. Hayat, and M. Ayub, "A substitution box generator, its analysis, and applications in image encryption," *Signal Processing*, vol. 187, Oct. 2021, doi: 10.1016/j.sigpro.2021.108144.