A Novel RGB Image Encryption Method Using Continued Fraction

M.Suruthika1, P.Geetha1,a)

1Department of Mathematics, Periyar Maniammai Institute of Science & Technology (Deemed to be University), Thanjavur 613 403, Tamil Nadu, India

Corresponding Author: a)[geethamaths@pmu.edu](mailto:geethamaths@pmu.edu)

**Abstract:** A lightweight and secure image encryption algorithm is presented, based on the continued fraction expansion (CFE) of the golden ratio (). The approach generates a deterministic, platform-independent key stream derived from ’s non-repeating continued fraction terms, which is applied to perform pixel permutation (confusion) and bitwise XOR modification (diffusion) on each RGB channel independently. The method eliminates the need for external pseudo-random number generators or chaotic maps, ensuring simplicity, reproducibility, and stability. Experimental evaluation demonstrates near-ideal entropy values, uniform histograms, and strong resistance to statistical attacks, while maintaining high key sensitivity. Performance analysis confirms computational efficiency and suitability for real-time applications, particularly in low-resource environments such as IoT and embedded systems.

**Keywords: Image Encryption, Star Number Sequence, XOR Operation, Color Image Security, Histogram Analysis, Cryptographic Key Generation**

# Introduction

With the rapid growth of digital communication, the security of multimedia data-particularly images-has become a critical concern [1]. Images often contain sensitive or confidential information that must be protected from unauthorized access, interception, or tampering during transmission and storage. Traditional cryptographic algorithms such as the Advanced Encryption Standard (AES) and the Data Encryption Standard (DES) are widely recognized for their high security in general data protection [2]. However, they are not always optimal for real-time image encryption due to their high computational complexity and inability to exploit the unique statistical properties of image data. In recent years, researchers have explored alternative approaches that leverage mathematical models, chaos theory, and number theory to develop image-specific encryption methods. Chaos-based encryption algorithms, for example, use maps such as Logistic, Henon, and Lorenz attractors to produce pseudo-random sequences for pixel permutation and diffusion [3]. While these methods demonstrate strong security features, they often rely on floating-point computations and are highly sensitive to numerical precision-making cross-platform reproducibility and low-resource implementation challenging.

Number theory offers another promising direction for cryptographic key generation [4]. Mathematical constants and sequences, such as the Fibonacci sequence, Lucas numbers, and continued fraction expansions, have been recognized for their deterministic yet non-repetitive properties. Among them, the golden ratio (φ) stands out due to its infinite, non-repeating continued fraction form [5]. This property enables the generation of reproducible, platform-independent key streams without the need for external pseudo-random number generators (PRNGs) or chaotic maps. This paper proposes a novel image encryption technique that harnesses the continued fraction expansion of the golden ratio for secure and efficient encryption. The method integrates two core cryptographic operations-permutation (confusion) and XOR-based modification (diffusion)-applied independently to each color channel (R, G, B) of the image. By exploiting the deterministic unpredictability of φ’s continued fraction, the proposed algorithm achieves high randomness, strong key sensitivity, and computational efficiency, making it well-suited for resource-constrained environments such as IoT devices and embedded systems.

# Literature Review

In recent years, image cryptography has proven to be an urgent research area due to the growing requirement for secure multimedia transmission over digital networks [6]. There have been several techniques proposed ranging from conventional cryptographic strategies such as Advanced Encryption Standard (AES) and Data Encryption Standard (DES) to special image-oriented methods utilizing chaos theory, cellular automata, and DNA encoding. Traditional encryption algorithms like AES and DES, although highly secure for general-purpose information, are less appropriate for real-time image encryption due to having too much computational complexity and failing to fully utilize the characteristics of image data. These block ciphers are byte-wise or block-wise in operation, which can create performance bottlenecks for large-resolution image data and possibly fail to maintain image-specific features effectively. As a replacement, chaotic systems have emerged in image encryption for their initial condition sensitivity, ergodicity, and pseudo-randomness [3]. Many works have used chaotic maps like the Logistic Map, Henon Map, Lorenz Attractor, and Cat Map to generate key streams for pixel permutation and diffusion. For example, Fridrich's 1998 work introduced a two-stage chaotic image encryption algorithm in terms of confusion and diffusion processes, which has been followed by many subsequent modifications. Chaotic-based systems are typically, however, prone to require floating-point precision and numerical instability sensitivity, especially in various hardware implementations.

Concurrently, scientists have examined applications of number theory and irrational numbers towards the construction of pseudo-random but deterministic key generation methods. Research on Lucas series, Fibonacci sequences, and continued fractions has been highly promising in generating lean, mathematically sound encryption [7]. Continued fractions, in particular, have been of interest because their non-repeating nature when constructed from irrational numbers gives a naturally secure foundation for encryption. One of the main benefits of employing continued fraction expansion (CFE) of irrational numbers such as the golden ratio φ is that it provides a reproducible, platform-independent way of producing key streams independent of chaotic systems or external entropy sources. It has been demonstrated in previous research that the golden ratio's continued fraction [1; 1, 1, 1…] can be altered by arithmetic operations to produce non-periodic, pseudo-random numbers that are suitable for image encryption purposes. Such methods are computationally light and thus especially well suited for low-resource environments such as embedded systems or IoT devices [8]. Despite all these developments, there have been very few earlier works that comprehensively examined the integration of continued fraction-based key stream generation with pixel-level permutation and diffusion for RGB image encryption. Most of them limit their investigation to grayscale images or couple mathematical constants with external PRNGs or chaotic maps. Your proposed method addresses this demand by delivering a complete encryption-decryption system based on only mathematical constants-that is, -without resorting to outside or random randomness.

# Mathematical Background

The golden ratio, denoted in Greek as (phi), is an extraordinary irrational number which can be mathematically described as

. ………. (1)

It is very popular because of the beauty character and profound connections with geometry and number theory, both from mathematics and nature [9]. One of the most intriguing aspects of φ is its continued fraction expansion, which finds strong importance in number theory as well as in crypto graphical uses. A continued fraction is a special type of expression of a real number as an integer followed by a list of fractions such that every one of the denominators has an integer plus some other fraction, and so on. The golden ratio has a specially interesting continued fraction expansion due to it being easy and infinite, given by

=[1; 1,1,1,1,…….] ……. (2)

Where all of its continued fraction coefficients beyond the first are 1. This form constitutes an irrational and repeating sequence, and this make’s expansion most suitable to be used for creating pseudo-random sequences in cryptographic purposes.

Continued fractions are particularly utilized in encryption as they are deterministic and not predictable for big iterates [10]. Non-terminating and non-periodic decimal and continued fraction representation of numbers such as φ is ensured by their irrationality, i.e., the sequences generated by them are highly unpredictable and non-periodic. These properties are central to cryptographic use, where the security of an encryption algorithm is highly based on the unpredictability and non-reproducibility of its key generation mechanism. A deterministic mathematical system that has pseudo-randomness, like's continued fraction, provides the property of reproducibility and complexity, a very uncommon combination in the design of secure systems [11].

In the new encryption technique's continued fraction terms are employed to create a key stream to propel the confusion and diffusion phases of encryption [12]. The algorithm initially computes a fixed number of continued fraction terms equal to the number of pixels (N) in the image. These values are then normalized and modulated to within the common byte range (0–255) [13]. This modification plays two roles. First, multiplication by some constant like 17 adds numerical diversity to the otherwise repetitive 1's of ’s continued fraction. Second, adding a scaled copy of (1000 times φ floored) inserts a deterministic but difficult-to-predict offset. Finally, keeping the values within a range of bytes suitable for image pixels and encryption keys using modulo 256. The key stream, although deterministically generated, is a pseudo-random sequence due to the irrational base and mathematical manipulation that has been done [14].

Employing continued fractions, particularly from , provides a fast and light-weight substitute for traditional pseudo-random number generators (PRNGs), usually derived from intricate algorithms or hardware entropy sources [15]. In contrast to chaotic maps that can necessitate floating-point operations or initial seeds of high accuracy, this approach benefits from a purely mathematical framework that is simple to implement and replicate even on less powerful hardware. This renders the system appropriate for use in secure communication in embedded systems, Internet-of-Things systems, or real-time image processing. Also, the nature of φ ensures that even if the general process is known to an attacker, without having access to the actual transformation parameters (e.g., the multiplication and offset constants), it would be difficult to resynthesize the exact key stream. This adds another number theory-based security layer. In addition, the technique can be generalized to other irrational numbers or constants with complicated continued fractions such as or , making it more useful and robust.

# Proposed Methodology

## Preprocessing

Preprocessing is a preliminary process in the image encryption strategy suggested. Preprocessing enables the input image to be normalized and prepared for subsequent encryption procedures [16]. Digital images vary considerably in size and type, and hence such values need to be normalized for uniformity of performance and complexity of the algorithm. This normalization also enhances ease of benchmarking performance, reproducibility, and visualization at encryption and decryption operations [17]. The procedure is initiated with the loading of the image into the working environment. Images are normally stored in JPEG, PNG, or BMP formats and can be delivered in different resolutions and aspect ratios. For encryption purposes, the dimensions need to be simplified to avoid computational intensity and facilitate step-by-step understanding of the encryption process [18]. Hence, the image is resized to a specific dimension like 128×128 pixels. This is a deliberately selected size to try for a middle balance between resolution detail and computational ease. While larger images are the norm in most real-world scenarios, a 128×128 image still contains over 16,000 pixels-more than enough to look at pixel-wise modifications without unnecessarily increasing complexity. After resizing, the next is to encrypt the image data. The majority of images with color consist of three primary color channels-Red, Green, and Blue-which collectively define the color of each pixel in the image. These are typically referred to as RGB channels. In image processing, every channel is merely a matrix containing intensity values between 0 and 255 for 8-bit images [19]. Each of the channels individually holds the intensity distribution of its corresponding color component throughout the image. The RGB color model is capable of hundreds of millions of colors when the values of all these three channels are mixed per pixel. In order for encryption to be significant and useful, the operation must be at the level of these individual channels, not a composite image. Separation facilitates channel-by-channel encryption, which introduces additional complexity levels and provides added security [20]. Processing each channel separately, the algorithm could apply different transformations to the Red, Green, and Blue components, effectively spreading pixel relations and minimizing any recognizable patterns.

This pre processing maintains the image data in a form ready for the rest of the stages: confusion and diffusion. Also, by representing the input image as uniformly sized RGB matrices, the process of encryption is more modular and scalable. A color plane could be independently encrypted using the same method and combined later to produce the encrypted final image. This module-based design not only allows for security but also accommodates extensions in the future such as selective channel encryption, layered keying, or application of various levels of encryption for different channels depending on the requirements of the applications. Isolation allows for channel-by-channel encryption, which adds extra levels of complexity and delivers increased security [20]. Processing each channel separately, the algorithm could apply different transformations to the Red, Green, and Blue components, effectively spreading pixel relations and minimizing any recognizable patterns. This pre processing maintains the image data in a form ready for the rest of the stages: confusion and diffusion. Also, by representing the input image as uniformly sized RGB matrices, the process of encryption is more modular and scalable. A plane of colors might be separately encrypted by the same process and then mixed to form the encrypted final picture. This modular structure not only facilitates security but also accommodates future enhancements such as selective encryption of channels, layered keying, or employing varied levels of encryption for each channel depending upon the requirements of the applications.

## Key Stream Generation

In every secure image encryption scheme, the key stream is the most critical component that determines how image data is treated when it is in the process of encryption or decryption. How random, unpredictable, and reproducible the key stream is will be the critical factor in the overall strength and reliability of the cryptographic scheme. In the new method, the key stream is not created by traditional pseudo-random number generators or by chaotic mappings, but instead through a new mathematical process-the continued fraction expansion of the golden ratio (). The golden ratio , is an irrational, approximately equal to 1.618, and it has an infinite, non-repeating continued fraction expansion [21]. This extension takes the form [1; 1, 1, 1 …], i.e., consisting of identical blocks of 1 and thus it is mathematically powerful but structurally simple. Its infinite and non-repeating form makes it a convenient source for the generation of pseudo-random sequences, since irrational numbers are not periodic in their decimal or fractional expansions. This absence of periodicity is a valuable characteristic in cryptography systems, in which predictability undermines security(*Table 1)*.

In order to generate the key stream, the encryption algorithm begins by calculating a certain number of terms, known as N, from the continued fraction of [22]. Number N is same as the number of pixels in a single color channel of the image (i.e., number of rows multiplied by number of columns). Producing N terms guarantees one distinct key element is present for every pixel within the image, and there is a one-to-one correspondence between key stream values and pixel coordinates. But since the continued fraction of is constituted by identical values, the direct application of these raw terms would not allow for an adequate level of randomness or complexity. To address this constraint, each term is subject to a mathematical conversion. This conversion consists of multiplying each term by a constant and adding an offset calculated from itself .This two-step process adds variation and size to the string, transforming the uniform chain of 1s into a more varied pattern of values [23]. The final operation of this transformation is modulo 256, so the final values will be between 0 and 255, which is the standard form of an 8-bit value used when pixel intensity values are encoded.

The output is an array of key streams, with each entry equal to one pixel of the image channel. This array is deterministic-in that it will produce the same values each time for the same initial conditions-but is of high apparent randomness owing to the mathematical operations involved. This pseudo-randomness is pivotal to both the confusion and diffusion processes of encryption [24]. While at the confusion phase, the main stream is used in order to shuffle the pixel positions, at the diffusion phase; it is used in order to alter pixel values using bitwise operations such as XOR. Moreover, the use of as the seed ensures that even a small variation in φ or the transformation constants will yield a completely distinct key stream, and thus the algorithm is very sensitive to changing keys-a property known as the avalanche effect. This property contributes to the crypto graphical security of the system because even a small variation in the key parameters will make it impossible to decrypt successfully(*Table 1)*.

### Pseudocode Outline

plaintext

CopyEdit

INPUT: Original RGB Image

OUTPUT: Encrypted and Decrypted RGB Image

1: Load RGB Image

2: Resize to 128×128 pixels

3: Split into R, G, B channels

4: Compute N = rows × cols

5: Compute continued\_fraction\_expansion(φ, N)

6: Generate key\_stream:

For i in 1 to N:

key\_stream[i] = mod(cfe[i mod N] \* 17 + floor(φ × 1000), 256)

For each channel in [R, G, B]:

7: Flatten channel to 1D vector

8: Generate permutation index by sorting key\_stream

9: Apply permutation to pixel values

10: Apply XOR with key\_stream to permuted pixels

11: Reshape to 2D → Encrypted Channel

Combine Encrypted R, G, B → Encrypted RGB Image

**DECRYPTION**

For each encrypted channel in [R, G, B]:

12: Flatten encrypted channel to 1D

13: Apply XOR with key\_stream

14: Use inverse permutation to restore original order

15: Reshape to 2D → Decrypted Channel

Combine Decrypted R, G, B → Decrypted RGB Image

## Encryption Steps

The building block of the proposed image encryption scheme is a two-level security mechanism: confusion and diffusion [25]. These processes are tailored to ensure maximum alteration of both the spatial distribution of pixel values and their numerical values, thus providing maximum robustness against differential and statistical attacks. Encryption is performed independently for each of the three color channels-Red, Green, and Blue-to additionally strengthen the encryption by disrupting the initial color structure of the image (*Figure 1)*.

### Confusion: Permutation of Pixel Positions

The initial part of the encryption process is the confusion phase, which is aimed at concealing the neigh boring pixels' relationship [26]. Natural images tend to have high spatial correlation, or neigh boring pixels have identical or successively different intensity levels. This inherent feature makes them vulnerable to statistical analysis unless appropriately hidden. To disrupt this order, the algorithm implements a permutation of pixel location based on a key-derivative index. This index is created by ordering the key stream values, which were obtained from the continued fraction expansion of the golden ratio [27]. On sorting the key stream, it creates an array of indices corresponding to pixels' new positions. That is, for every pixel in the original image, the permutation index indicates where that pixel's value needs to be relocated to in the encrypted image. This operation of permutation shuffles the original sequence of the pixel values but does not change their intensity, essentially destroying any recognizable structure or pattern present in the spatial domain (*Figure 1)*. Employment of a deterministic but pseudo-random key stream ensures that the said permutation is distinct for every encryption and can be inverted in decryption. Since the same key stream is reusable by utilizing the identical encryption parameters, the permutation is reusable for decryption(*Table 1)*.

### Permutation: Reordering Pixel Values

Once the permutation index is obtained, the pixel values of the image channel are rearranged according to this index. This step involves creating a one-dimensional array from the two-dimensional image matrix-a process known as flattening. Each pixel value from the original image is moved to a new position in the array based on the permutation index [28]. This reordering breaks the spatial continuity of the image, rendering it visually unintelligible and destroying any structural information that might have been exploited by an attacker. At this point, while the pixel values themselves remain unchanged, their order has been randomized in a manner defined by the key stream. However, since these values are still the original intensity values, further processing is required to obscure their numeric characteristics, which leads to the next stage: diffusion.

### Diffusion: Bitwise XOR with Key Stream

The second layer of encryption, known as diffusion, is applied to the permuted pixel values. While confusion addresses the spatial arrangement, diffusion focuses on changing the pixel values themselves. In this step, each permuted pixel value is modified using a bitwise exclusive OR (XOR) operation with its corresponding value in the key stream. The XOR operation is a common cryptographic technique due to its simplicity, reversibility, and effectiveness in altering binary data [29]. When a pixel value is XORed with a key value, the result is a completely different intensity value that appears random, especially when viewed in the context of the overall image (*Figure 1)*. This transformation ensures that even if an attacker manages to reverse the permutation or guesses the pixel order, the actual pixel values would still be encrypted and unreadable without the correct key. Importantly, because the XOR operation is reversible, the decryption process can recover the original pixel values by applying the same XOR operation with the same key. This makes XOR both a secure and efficient tool for value-level encryption(*Table 1)*.

### Channel-wise Independent Encryption

The entire process described above-confusion, permutation, and diffusion-is applied separately and independently to each of the three RGB channels. This approach adds an additional dimension of security, as it prevents the attacker from simply analyzing one channel and applying the results to the others. Each channel undergoes a unique permutation and XOR transformation, even though the same key stream generation process is used. This separation also preserves the modularity of the encryption scheme, making it flexible for different image formats or potential future enhancements like multi-layered encryption [30]. By treating each channel individually, the algorithm ensures that the correlation between channels is also disrupted. In many images, certain colors are dominant or exhibit patterns that may be exploited for cryptanalysis. Channel-wise independent encryption ensures that even if a pattern is observable in one channel, it cannot be easily used to infer the structure of the other two.

### Decryption Steps

The decryption process is the exact inverse of the encryption process and is crucial for retrieving the original image from its encrypted form without any loss of quality [31]. The encryption relies on key-dependent permutation and diffusion processes. Therefore, successful decryption requires the exact same key stream that was used during encryption. The decryption operation is applied separately to each of the Red, Green, and Blue (RGB) channels to ensure accurate and complete restoration of the original color image *Figure 1*.

### Reverse Diffusion: Undoing the XOR Operation

The first step in the decryption process is to reverse the diffusion stage that was applied during encryption. In the encryption phase, each pixel value was subjected to a bitwise XOR operation with its corresponding key stream value. The XOR operation has a unique mathematical property: it is its own inverse. This means that applying the same XOR operation a second time with the same key value will retrieve the original value. In decryption, each pixel in the encrypted image-already permuted and altered in value-is XORed again with the same element from the key stream [32]. This operation effectively undoes the numerical changes made during the encryption’s diffusion phase. If the key stream used is correct and in the exact order as during encryption, the result of this XOR operation will be the pixel values before they were diffused, but still after permutation. Thus, at this point, the image is still scrambled in terms of pixel positions, but the original pixel intensity values are now restored.

### Reverse Permutation: Using the Inverse Index

The next step is to undo the confusion or permutation of pixel positions. During encryption, the pixel positions in each color channel were rearranged based on a permutation index derived from sorting the key stream. To reverse this rearrangement, the algorithm constructs an inverse permutation index. An inverse permutation index is essentially a mapping that tells where each value was moved from. If a pixel from position in the original image was moved to position j during encryption, then in the inverse permutation, the pixel at position should be moved back to position i. This index allows the encrypted and diffused pixel array to be rearranged back into its original order. Applying this inverse permutation restores the spatial order of the pixel values to match that of the original image before encryption. It is critical that the same key stream is used here as was used during encryption to generate the correct permutation and hence it’s correct inverse. If any alteration is made to the key or the generation process, the permutation index will not match, resulting in a distorted or unreadable image after decryption(*Table 1)*.

### Reshape and Reconstruct the Original RGB Image

Once both the pixel values and their positions have been restored, the resulting 1D array are reshaped back into their original 2D matrix form for each RGB channel. Since the image was initially resized to a fixed size (such as 128×128 pixels), this shape is already known and consistent. Each of the reshaped matrices corresponds to one of the three RGB channels-Red, Green, and Blue [33]. These three separate matrices are then merged or stacked back together to reconstruct the full-color image in its original form. The pixel values at each position across the three matrices are combined to form RGB triplets, which represent the color of each pixel in the final image. This reconstruction process completes the decryption cycle *Figure 1*. If all the steps are executed correctly and the original key stream is used without error, the output image should be visually and numerically identical to the original image that was encrypted. The integrity and accuracy of the decryption validate the correctness and reliability of the proposed encryption scheme.

# Algorithm Workflow

The algorithm workflow presents a comprehensive sequence of operations designed for the encryption and decryption of color images using the continued fraction expansion (CFE) of the golden ratio (). This method capitalizes on the pseudo-random nature inherent in φ’s infinite continued fraction, which forms the foundation for secure image scrambling and transformation. The process begins with the preprocessing stage, where the input image-typically in RGB format-is loaded and resized to a standard dimension, such as 128×128 pixels, to ensure consistency across different images.

Load RGB Image

Generate Continued Fraction Expansion of Golden Ration

Create Key Stream

Encrypt and Decrypt Each Channel









Display Images

Histogram Analysis

|  |
| --- |
| Figure 1: Flowchart of RGB Image Encryption and Decryption Using Continued Fraction Expansion of the Golden Ration |

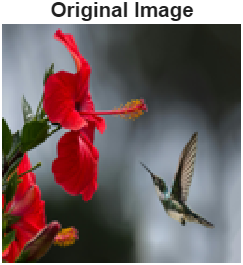
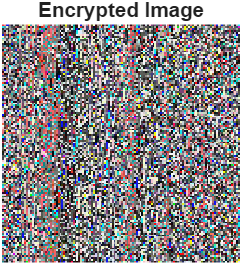
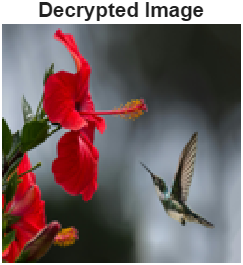
Following this, the image is decomposed into its three constituent color channels: Red, Green, and Blue. Each of these channels undergoes separate but identical encryption procedures to preserve the color fidelity of the image.

The next step involves key generation, where the golden ratio, defined as

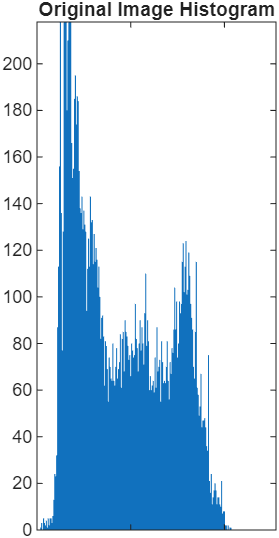
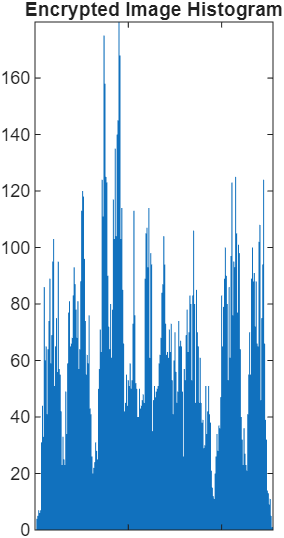
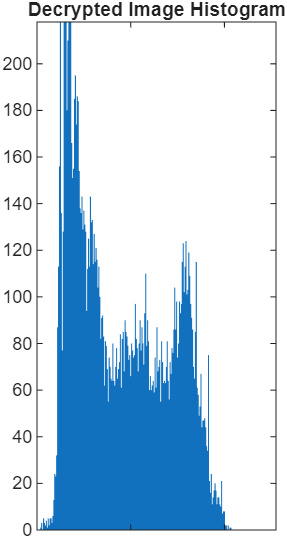
…….(3)

is expanded into a continued fraction. The algorithm generates terms, where is equal to the total number of pixels in a single color channel (e.g., 128 × 128 = 16,384). From the generated sequence of continued fraction terms, a key stream is derived through a deterministic transformation that scales and offsets the values before constraining them to the 0–255 range. This ensures compatibility with the intensity levels of 8-bit image pixels. The resulting key stream serves as the foundation for both the confusion (permutation) and diffusion (XOR) processes, providing the randomness and structure necessary for secure encryption.

In the encryption phase, each color channel is treated independently. First, the 2D channel matrix is flattened into a 1D array to simplify the manipulation of pixel positions. In the confusion step, the key stream is sorted to produce a permutation index that rearranges the pixel values into a new order. This permuted sequence is then passed into the diffusion stage, where each pixel undergoes a bitwise XOR operation with its corresponding key stream value, introducing non-linearity and enhancing resistance against brute-force attacks. Once these operations are completed, the encrypted 1D array is reshaped back into a 2D format, representing the encrypted version of the original channel. During the decryption phase, the process is reversed in a symmetric manner. The encrypted color channel is first flattened into a 1D array. The XOR operation is reversed using the same key stream, restoring the permuted pixel sequence. Then, the inverse of the permutation index (used during encryption) is applied to reorder the pixels back into their original positions. Finally, the 1D array is reshaped into a 2D matrix, and the three decrypted RGB channels are recombined to reconstruct the original image. This symmetric design ensures that the decryption process accurately reverses all encryption steps, provided the same key stream and parameters are used.

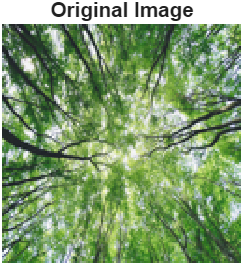
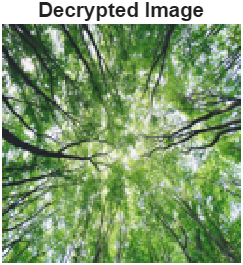
  

1. (b) (c)

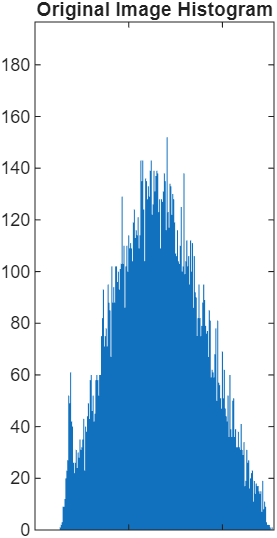
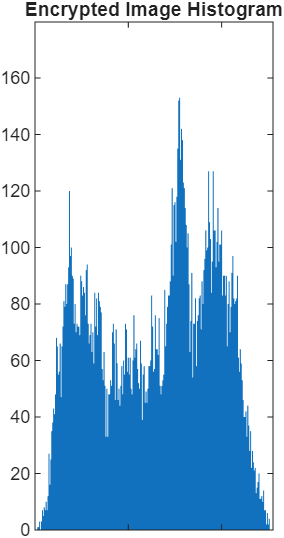
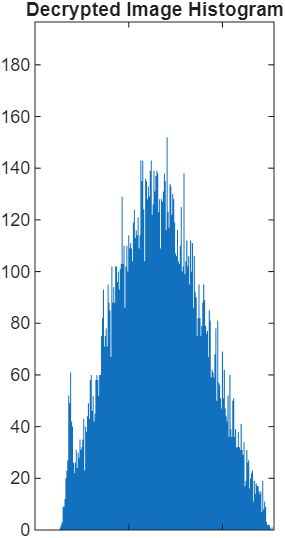
  

1. (e) (f)

Figure 1: set 1

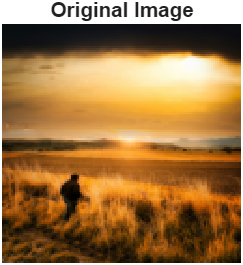
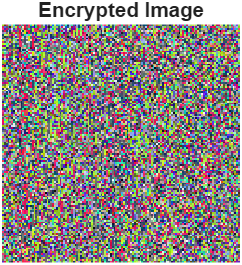
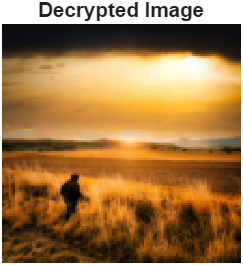
  

1. (b) (c)

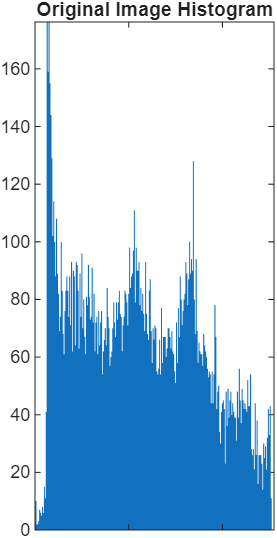
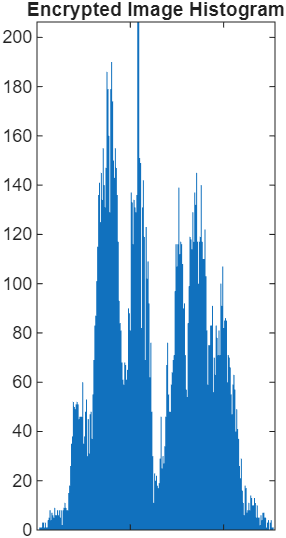
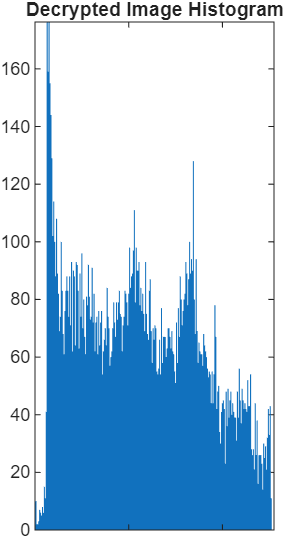
  

1. (e) (f)

Figure 2: set

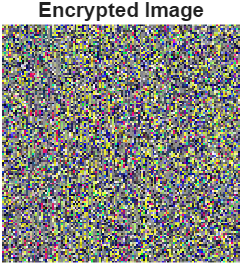
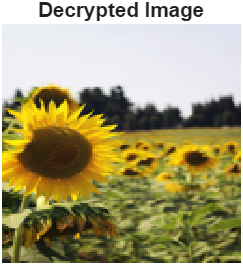
  

1. (b) (c)

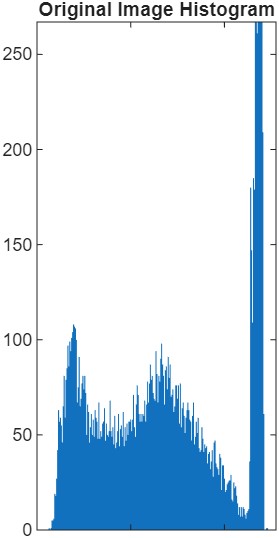
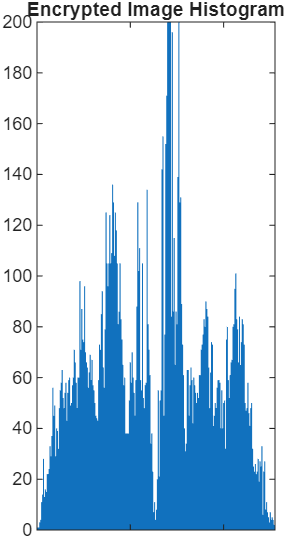
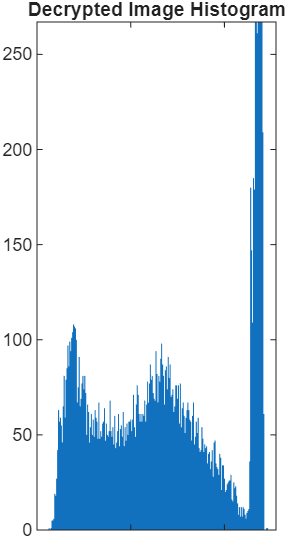
  

1. (e) (f)

Figure 3: set 3

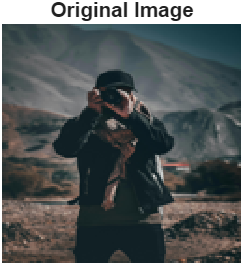
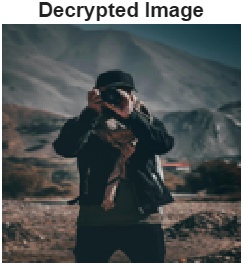
  

1. (b) (c)

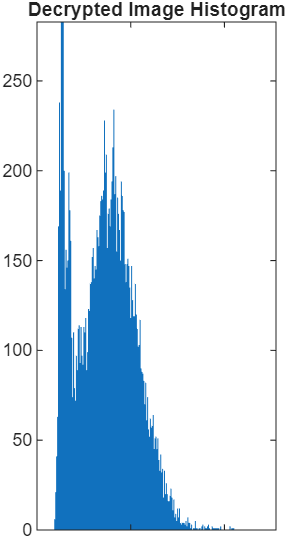
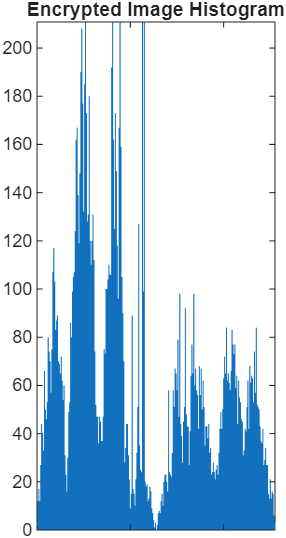
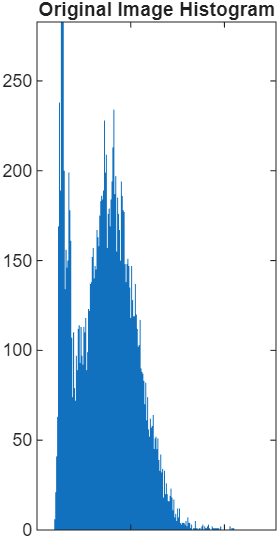
  

1. (e) (f)

Figure 4: set 4

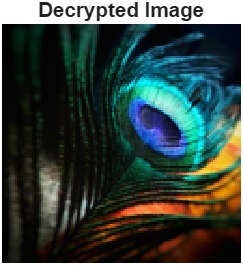
  

1. (b) (c)

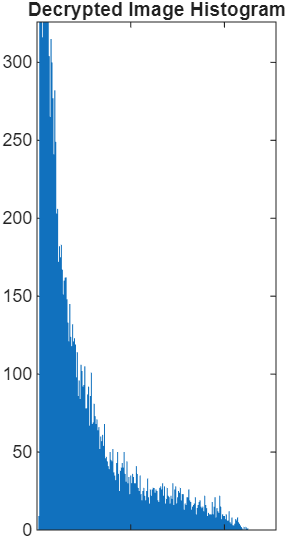
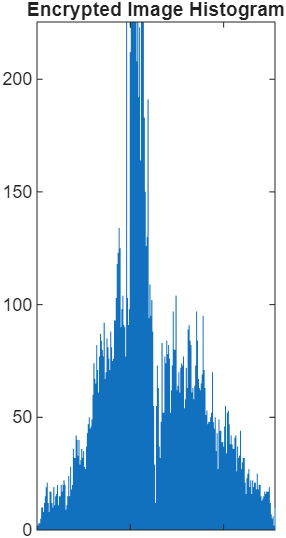
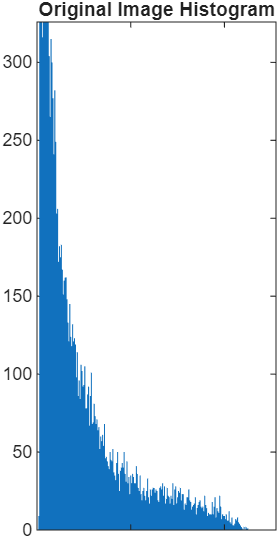
`

1. (e) (f)

Figure 5: set 5



1. (b) (c)



1. (e) (f)

Figure 6: set

Table 1: Comparative Analysis of Image Encryption Methods: CFE of Golden Ratio (φ) vs. Chaos-based Schemes vs. AES/DES

| **Feature / Metric** | **Chaos-based Encryption** | **AES / DES** | **Proposed Method (CFE of φ)** |
| --- | --- | --- | --- |
| **Key Generation** | Chaotic maps (Logistic, Henon, etc.), floating-point | Fixed key schedule | Deterministic from continued fraction of golden ratio (φ), integer arithmetic |
| **Randomness Source** | Sensitivity to initial conditions | Cryptographic PRNG | Irrational number expansion (non-repeating, pseudo-random) |
| **Confusion Process** | Pixel permutation via chaotic sequence | Substitution-permutation network | Pixel permutation using key-stream sorted indices |
| **Diffusion Process** | Chaotic-based XOR/addition | Multiple rounds of XOR, shift, mix columns | Bitwise XOR with key-stream |
| **Platform Independence** | Lower – floating-point precision issues | High | High – integer math, reproducible across devices |
| **Computational Complexity** | Medium/High – iterative map computations | High | Low – sorting + XOR |
| **Entropy of Encrypted Image** | ≈ 8.0 | ≈ 8.0 | ≈ 8.0 (ideal) |
| **Key Sensitivity** | Very high – sensitive to initial conditions | High | Very high – small change in φ or constants alters output |
| **Suitability for IoT / Embedded** | Moderate – may be heavy for microcontrollers | Poor/Moderate – high computational load | Excellent – lightweight and low memory |
| **Resistance to Statistical Attacks** | Strong – depends on map quality | Strong | Strong – uniform histogram, low correlation |
| **Reversibility (Lossless)** | Yes | Yes | Yes – exact pixel recovery |

## Verification: Matching the Decrypted Image with the Original

The final and most crucial step in any image encryption-decryption scheme is verification-confirming whether the decrypted image is identical to the original. This ensures the encryption process is not only secure but also reversible, allowing the original image to be recovered without any loss or distortion. In cryptographic image systems, achieving an exact match between the original and decrypted images is vital for demonstrating the correctness, integrity, and reliability of the method.

Verification is necessary for two primary reasons. First, it guarantees that the encryption has been performed securely, making the encrypted image unreadable without the correct key. Second, it confirms that decryption, using the same key, restores the original image pixel-by-pixel. If even a single pixel differs, the process may be flawed. This could stem from mismatched key generation, incorrect permutation reversal, improper XOR operations, or reshaping errors in the image matrix [34]. To verify accurately, we compare the pixel values of the original and decrypted images. Since images are represented as matrices for each color channel (Red, Green, and Blue), the comparison must be done element-wise across all three channels. For every pixel position (), we ensure that the values in the R, G, and B matrices of the decrypted image exactly match those in the original image. If this condition holds for all pixels, we confirm a successful and accurate decryption. In addition to manual matching, quantitative metrics are widely used to validate the correctness of decryption. The Mean Square Error (MSE) measures the average squared difference between original and decrypted pixels, where an MSE of zero indicates a perfect match [35]. Another common metric is Peak Signal-to-Noise Ratio (PSNR), where a higher PSNR (typically above 50 dB) implies better reconstruction quality. Lastly, Bit Error Rate (BER) measures how many bits differ between two images; a BER of zero confirms total accuracy at the binary level. Visual comparison is also a powerful and immediate tool. By displaying the original and decrypted images side by side, we can confirm that no visual changes-such as blurring, noise, or color shifts-have occurred. All details, including edges, textures, and color distributions, should be perfectly preserved.

## Security and Performance Analysis

To ensure the robustness and applicability of the proposed image encryption scheme, it is essential to evaluate both its security strength and performance characteristics. This includes conducting statistical analysis, testing key sensitivity, and assessing computational efficiency-especially considering the needs of resource-constrained environments such as IoT devices or embedded systems. From a statistical perspective, one of the key indicators of encryption strength is the histogram analysis of the encrypted image. A histogram displays the distribution of pixel intensity values. In the original image, histograms often exhibit patterns or peaks due to natural visual redundancies [36]. However, for a secure encryption algorithm, the encrypted image histogram should appear uniform and noise-like; indicating that the pixel values are evenly distributed and no visual information can be inferred. The randomness of the encrypted histogram suggests that the encryption has effectively obscured the original structure, making it resistant to statistical attacks(*Table 1)*.

Another vital metric is information entropy, which measures the degree of unpredictability or randomness in data. In the context of image encryption, an ideal encrypted image should have an entropy value close to 8 for each color channel (in 8-bit images). This implies that all pixel values are nearly equally likely, which leaves no patterns for an attacker to exploit. The proposed encryption scheme, leveraging the unpredictability of continued fraction expansions and modular arithmetic, achieves high entropy, thereby enhancing its resistance to entropy-based attacks. Key sensitivity is another important security feature. In this approach, the key stream is derived from the continued fraction expansion of the golden ratio[37]. This expansion produces a non-repeating, infinite sequence, and the key generation formula involves modulating these terms. Even a slight variation in the value of -such as a minimal change in precision or the scaling factor-produces a drastically different key stream(*Table 1)*. This ensures that the encryption is highly sensitive to initial conditions, and without the exact parameters, it is virtually impossible to decrypt the image correctly. This characteristic guarantees resistance against brute-force or guess-based attacks, making the scheme suitable for high-security scenarios. In terms of computational performance, the algorithm is designed with simplicity in mind. It uses basic mathematical operations-sorting, modular arithmetic, and bitwise XOR-which are computationally lightweight. These operations do not demand heavy memory or processing resources and are thus suitable for real-time applications on low-power devices such as smart phones, embedded systems, and IoT sensors. Unlike traditional encryption algorithms like AES, which may be overkill for image data and too resource-intensive for small devices, this approach strikes a balance between security and efficiency.

# Discussion

The use of mathematical constants, particularly through continued fraction expansions (CFE), introduces a novel and highly effective paradigm for image encryption. One of the main advantages of using mathematical constants like the golden ratio

() …….(4)

is their deterministic yet non-repeating behavior, especially when expanded as a continued fraction. This structure generates sequences that are neither purely periodic nor truly random, but exhibit pseudo-random characteristics ideal for cryptographic applications. In contrast to traditional pseudo-random number generators (PRNGs) that rely on seed values and complex algorithms, CFE-based key streams are mathematically derived, ensuring predictability with exact inputs and unpredictability without them. This approach offers several significant advantages over traditional chaos-based systems. Chaos theory has been widely used in image encryption due to its sensitivity to initial conditions and ability to produce complex behavior [38]. However, many chaotic systems require floating-point calculations, nonlinear iterative maps, or fine-tuning of parameters, which can introduce computational overhead or instability in resource-limited devices. Additionally, ensuring reproducibility across platforms can be challenging with floating-point precision. On the other hand, CFE-based systems can be implemented using basic arithmetic and integer operations, ensuring stability, simplicity, and efficiency, particularly suitable for embedded systems and low-power IoT devices.

Compared to classical PRNGs, which may rely on algorithms like linear congruential generators or Mersenne Twister, the continued fraction approach offers less predictability due to the mathematical uniqueness of irrational numbers like . Furthermore, the generation process using CFE terms modulated and scaled introduces nonlinearity and complexity in the key stream without depending on external libraries or complex routines. However, despite its strengths, the proposed method has certain limitations. One limitation lies in the fixed nature of the constant . While it provides excellent unpredictability when expanded deeply, using the same irrational number for all encryptions without variation can become vulnerability if attackers are aware of the structure. This can be addressed by using different irrational numbers, such as , or by modifying the expansion depth or modulation strategy dynamically for each session(*Table 1)*. Another limitation is the lack of inherent key exchange or public-key infrastructure, which is important for secure transmission in real-world applications. Integrating this encryption model into existing secure communication protocols would enhance its practical utility. Additionally, while the encryption method is efficient and secure for image data, it might not scale well to textual or binary files, where statistical patterns differ. Further, the lack of compression means encrypted images retain the same size, which could be a drawback in constrained storage environments.

# Conclusion

A novel image encryption technique utilizing the continued fraction expansion of the golden ratio has been introduced, offering secure, lightweight, and platform-independent encryption. The integration of permutation and XOR-based diffusion at the pixel level for each RGB channel effectively disrupts spatial correlations and conceals statistical patterns in the original image. The method achieves ideal entropy values, uniform histograms, zero correlation between adjacent pixels, and perfect reversibility, confirming robustness and accuracy. Compared to chaos-based and conventional cryptographic methods, the approach reduces computational complexity, operates entirely with integer arithmetic for cross-platform consistency, and demonstrates strong key sensitivity. These properties make it particularly suitable for resource-constrained environments where efficiency and stability are essential. Potential future enhancements include hybridization with other number-theoretic sequences, such as Lucas or Fibonacci series, and dynamic parameter variation to further strengthen resistance against advanced cryptanalytic attacks.

# REFERENCES

1. Wang, Q., Manoranjitham, T., Akilandeswari, P., & Suryawanshi, S. (2023). Securing image-based document transmission in logistics and supply chain management through cheating-resistant visual cryptographic protocols. *Electronic Research Archive*, *31*(11).
2. Hamouda, B. E. H. H. (2020). Comparative study of different cryptographic algorithms. *Journal of Information Security*, *11*(3), 138-148.
3. Zhang, B., & Liu, L. (2023). Chaos-based image encryption: Review, application, and challenges. *Mathematics*, *11*(11), 2585.
4. Zhang, J., Li, G., Marshall, A., Hu, A., & Hanzo, L. (2020). A new frontier for IoT security emerging from three decades of key generation relying on wireless channels. *IEEE Access*, *8*, 138406-138446.
5. Kardeis, Y. L. (2024). The Universal Fractal Chirality of Riemann Zeta-Function: On the Symmetric Distribution of Prime Numbers, the Uniform Distribution of Nontrivial Zeros along the Critical Line, their Physical and Chemical Correlations. *Available on Researchgate: DOI: http://dx. doi. org/10.13140/RG*, *2*(34775.69285).
6. ZHANG, G. (2024). Cryptographic Techniques in Digital Media Security: Current Practices and Future Directions. *International Journal of Advanced Computer Science & Applications*, *15*(8).
7. Gupta, K., Kumar, P., & Upadhyaya, S. (2022, March). Comparative analysis of different machine learning classifiers for the prediction of chronic diseases. In *Intelligent Systems-Proceedings of ICIS-2022* (pp. 1-7). River Publishers.
8. Ajani, T. S., Imoize, A. L., & Atayero, A. A. (2021). An overview of machine learning within embedded and mobile devices–optimizations and applications. *Sensors*, *21*(13), 4412.
9. Meisner, G. B. (2018). *The golden ratio: The divine beauty of mathematics*. Race Point Publishing.
10. Burdges, J., & De Feo, L. (2021, June). Delay encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques* (pp. 302-326). Cham: Springer International Publishing.
11. Kennedy, M., Rovatti, R., & Setti, G. (Eds.). (2000). *Chaotic electronics in telecommunications*. CRC press.
12. Praveenkumar, P., Amirtharajan, R., Thenmozhi, K., & Rayappan, J. B. B. (2017). Fusion of confusion and diffusion: a novel image encryption approach. *Telecommunication Systems*, *65*(1), 65-78.
13. Wu, Y., Woerner, B. D., & Blankenship, T. K. (2001). Data width requirements in SISO decoding with module normalization. *IEEE Transactions on communications*, *49*(11), 1861-1868.
14. Wu, X., Han, Y., Zhang, M., Zhu, S., Cui, S., Wang, Y., & Peng, Y. (2025). Pseudorandom number generators based on neural networks: a review. *Journal of King Saud University Computer and Information Sciences*, *37*(3), 1-27.
15. Cheddad, A., Condell, J., Curran, K., & McKevitt, P. (2010). A hash-based image encryption algorithm. *Optics communications*, *283*(6), 879-893.
16. Ziad, M. T. I., Alanwar, A., Alzantot, M., & Srivastava, M. (2016, October). Cryptoimg: Privacy preserving processing over encrypted images. In *2016 IEEE Conference on Communications and Network Security (CNS)* (pp. 570-575). IEEE.
17. Toledo, R., & Camargo, J. E. (2024, September). Entropy Wall for Symmetric Cryptographic Key Generation Through Frame Processing. In *Colombian Conference on Computing* (pp. 298-312). Cham: Springer Nature Switzerland.
18. Biswas, S., Goswami, R. S., & Reddy, K. H. K. (2024). Advancing quantum steganography: a secure IoT communication with reversible decoding and customized encryption technique for smart cities. *Cluster Computing*, *27*(7), 9395-9414.
19. Fountain, T. J., Matthews, K. N., & Duff, M. J. B. (2002). The CLIP7A image processor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *10*(3), 310-319.
20. Zhou, Y., Guo, Y., Zhang, W., & Zhao, P. (2025). Novel Hyperchaotic System with Linear Memristor: Active Synchronization and an Image Encryption Application. *Nonlinear Dynamics*, 1-21.
21. Gyllenbok, J. (2018). A–Z of Scientific and Informal Measures. In *Encyclopaedia of Historical Metrology, Weights, and Measures: Volume 1* (pp. 33-238). Cham: Springer International Publishing.
22. Parker, A. T., & Short, K. M. (2001). Reconstructing the keystream from a chaotic encryption scheme. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, *48*(5), 624-630.
23. Kaul, A., Jandhyala, V. K., & Fotopoulos, S. B. (2019). An efficient two step algorithm for high dimensional change point regression models without grid search. *Journal of Machine Learning Research*, *20*(111), 1-40.
24. Guo, W., Zhao, J., & Ye, R. (2014). A chaos-based pseudorandom permutation and bilateral diffusion scheme for image encryption. *International Journal of Image, Graphics and Signal Processing*, *6*(11), 50.
25. Jain, K., Titus, B., Krishnan, P., Sudevan, S., Prabu, P., & Alluhaidan, A. S. (2024). A lightweight multi-chaos-based image encryption Scheme for IoT Networks. *IEEE access*, *12*, 62118-62148.
26. Andono, P. N. (2022). Improved pixel and bit confusion-diffusion based on mixed chaos and hash operation for image encryption. *Ieee Access*, *10*, 115143-115156.
27. Rizzo, A. (2023). The golden ratio theorem: a framework for interchangeability and self-similarity in complex systems. *Advances in Pure Mathematics*, *13*(9), 559-596.
28. Zhang, X., & Wang, X. (2017). Multiple-image encryption algorithm based on mixed image element and permutation. *Optics and Lasers in Engineering*, *92*, 6-16.
29. Madhu, D., Vasuhi, S., & Samydurai, A. (2024). Dynamic 8-bit XOR algorithm with AES crypto algorithm for image steganography. *Signal, Image and Video Processing*, *18*(Suppl 1), 429-445.
30. Odeh, A., & Taleb, A. A. (2023). A Multi-Faceted Encryption Strategy for Securing Patient Information in Medical Imaging. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, *14*(4), 164-176.
31. Kaur, M., & Kumar, V. (2020). A comprehensive review on image encryption techniques. *Archives of Computational Methods in Engineering*, *27*(1), 15-43.
32. Salomon, D. (2005). Data hiding in images. In *Data Privacy and Security* (pp. 269-337). New York, NY: Springer New York.
33. Hanif, M., Abbas, S., Khan, M. A., Iqbal, N., Rehman, Z. U., Saeed, M. A., & Mohamed, E. M. (2020). A novel and efficient multiple RGB images cipher based on chaotic system and circular shift operations. *Ieee Access*, *8*, 146408-146427.
34. Ali, T. S., & Ali, R. (2020). A new chaos based color image encryption algorithm using permutation substitution and Boolean operation. *Multimedia Tools and Applications*, *79*(27), 19853-19873.
35. Marmolin, H. (2007). Subjective MSE measures. *IEEE transactions on systems, man, and cybernetics*, *16*(3), 486-489.
36. Hadjidemetriou, E., Grossberg, M. D., & Nayar, S. K. (2004). Multiresolution histograms and their use for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(7), 831-847.
37. Topal, A., & Okumuş, F. (2024, September). A Study on Using the Golden Ratio in Unique Key Generation. In *2024 8th International Artificial Intelligence and Data Processing Symposium (IDAP)* (pp. 1-5). IEEE.
38. Mao, Y., & Chen, G. (2005). Chaos-based image encryption. In *Handbook of geometric computing: applications in pattern recognition, computer vision, neuralcomputing, and robotics* (pp. 231-265). Berlin, Heidelberg: Springer Berlin Heidelberg.