

RGB IMAGE ENCRYPTION ALGORITHM USING 6D HYPERCHAOTIC SYSTEM AND FIBONACCI Q-MATRIX

HUSSEIN YOUNIS¹, MUJAHED ELEYAT²

^{1,2}Department of Natural, Engineering and Technology Sciences, Arab American University, Palestine

E-mail: ¹Hussein.Younis@aaup.edu, ²mujahed.eleyat@aaup.edu

ABSTRACT

The advancement of ICT and the widespread use of the Internet enables users to store many types of data via the Internet, and thus these data may be vulnerable to hacking or illegal access. As a result, certain mechanisms must be designed to guarantee that these data are secured. Encryption is one of the most significant methods of image protection as it's based on generating a different image in terms of content from the original image, making it difficult to identify or retrieve the original image only through a key. In this work, an RGB Image Encryption Algorithm Using a 6D Hyperchaotic System and Fibonacci Q-matrix is presented. Confusion and diffusion are used to generate the encrypted image to achieve a high-security level. The proposed encryption algorithm was tested against different types of noise and attacks. Moreover, we combined the proposed algorithm against other algorithms in terms of entropy. The result shows that the proposed algorithm outperformed the existing algorithms. Also, the result shows that the speed up in the execution time of the proposed algorithm can be enhanced up to 2.23x faster by employing parallel programming.

Keywords: RGB Image Encryption, 6D Hyperchaotic Chen System, Fibonacci Q-Matrix, Parallelism, Multiprocessing.

1. INTRODUCTION

Images are widely used in communication around the world. Images are used in many applications including medical, remote sensing, educational imaging, e-commerce, and more. With the convenience of the Internet, we can share images and information anytime, anywhere.

However, some images may refer to private personal information or other commercial information. Image security should be given high priority to protect against unauthorized access, noise, and other attacks. There are different approaches to image security, one of the most known is image encryption which is based on generating another image from the original image, which is called the encrypted image, where this encrypted image is different in terms of content and hence the difficulty of identifying the original image through it except with the key [1]. Only the owner of the original image can restore it using the key as depicted in Figure 1.

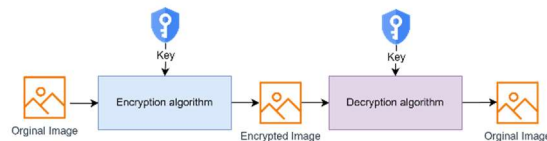


Figure 1: Cryptographic Algorithm

When the same key is used for both encryption and decryption, it is referred to as a symmetric key. On the other hand, if different keys are used for encryption and decryption, it is known as an asymmetric key [2].

Researchers have introduced numerous encryption algorithms that vary in terms of the type and size of data that can be encrypted, the number and size of keys required, and whether symmetric or asymmetric keys are utilized.

In the context of Image encryption, the encryption algorithm involves the properties of confusion and diffusion to achieve security, confusion means changing an image pixel's position without changing its value and diffusion refers to changing the individual pixel image value [3].

This paper presents a new RGB image encryption algorithm using a 6D hyperchaotic system and Fibonacci Q-matrix. The main

contributions of this paper are: (1) A 6D fractional-order hyperchaotic Chen system is generated and used to scramble each channel of the original RGB image (2) Fibonacci Q-matrix is used for image diffusion (3) Enhance the proposed algorithm performance using multiprocessing.

The main contributions of this paper are as follows:

1. Development of a robust encryption algorithm capable of withstanding various types of attacks, including brute force attacks and vulnerabilities.
2. Integration of the 6D Hyperchaotic System into the proposed encryption algorithm.
3. Encryption of RGB images, offering a convenient method for file encryption and decryption.
4. The utilization of multiprocessing in the implementation of the encryption algorithm represents a significant advancement in enhancing the algorithm's efficiency and performance.

The following sections are The related work presented in Section 2. The preliminaries presented in Section 3. The proposed encryption and decryption algorithm is presented in Sections 4 and 5. Experiments and results are discussed in Section 6. Performance enchantment for the proposed algorithm is presented in section 7. The comparison between the proposed algorithm and other algorithms is presented in section 8. The final section concludes this paper.

2. RELATED WORK

In recent years, many image encryption algorithms have been proposed and published for greyscale and colored images. These encryption algorithms are inspired by many fields such as the theory of chaos, DNA strand construction, deep neural network (DNN), and plaintext-related shuffling. Despite the diversity of these algorithms, there are still some dilemmas that are summarized in (1) Algorithms' performance and how to take advantage of ICT development to improve performance such as parallelism (2) The ability of encryption algorithms to retrieve the original image despite cuts or noise in the encrypted image (3) Percentage of algorithm's protection against exposure to various types of attacks.

In [3] authors produce a grey image encryption algorithm using a 6D hyperchaotic system and Fibonacci Q-matrix. They achieved a high level of security by applying double confusion and diffusion.

They produce an experimental test against a different type of attack and noise, and the result that they obtained presents the reliability and effectiveness of their proposed algorithm. The time complexity for their algorithm was $O(M \times N)$ where M represents the number of image rows and N represents the number of image columns. However, an important consideration arises regarding the exclusion of colored image encryption in the proposed algorithm. Given the prevalence of colored images in modern applications, the absence of color encryption capabilities may restrict the algorithm's practical utility in diverse scenarios.

The authors of [4] present a color image encryption algorithm using the fractional-order hyperchaotic system, they use a 4D hyperchaotic Chen system with Fibonacci Q-matrix for confusion and diffusion to achieve a high level of security. The algorithms proposed by them are based on splitting the image into three channels (red, green, and blue) and then applying confusion and diffusion on each channel to get the encrypted image. They perform experimental testing to ensure the efficiency and ability to resist the attack of the proposed algorithm. However, A 6D system offers higher complexity and more dimensions compared to a 4D system, potentially providing increased security through a higher degree of chaotic behavior.

The authors in [5] present an encryption algorithm that combines between 3d hyperchaotic Chen system and stack data structure to encrypt the image. They applied extensive simulations to ensure the efficiency and security of the proposed algorithm. The result they obtained shows that the entropy value for the decrypted image is close to the entropy value for the original image but not identical. They compare the proposed algorithm with other algorithms based on entropy value and they achieve higher values than others [6] [7] [8]. However, the usage of queues and trees can offer benefits in organizing and processing data, their use in image encryption may come with drawbacks related to complexity, memory usage, processing overhead, algorithmic efficiency, security implications, and scalability.

Other researchers use the DNA strand construction to produce an encryption algorithm for the colored image. Authors in [9] produce an encryption algorithm by assigning image pixels according to predefined DNA bases to generate DNA binary strings. The result is converted to a hexadecimal number and performed XOR with predefined values to get a decimal value. Then an e-function matrix is generated based on previous

values to obtain the encrypted image. They perform experimental testing to measure the quality and accuracy of the proposed algorithm.

In light of these existing works, our research introduces a novel RGB image encryption algorithm leveraging a 6D hyperchaotic system and Fibonacci Q-matrix for enhanced security and performance. By generating a 6D fractional-order hyperchaotic Chen system and implementing multiprocessing techniques for algorithm optimization, our approach aims to address the limitations of previous algorithms while ensuring robust encryption of colored images. Through this comparative analysis, we aim to contribute to the advancement of image encryption techniques and provide insights into improving algorithm efficiency and security.

3. PRELIMINARIES

Digital images are represented using different types of color space. The most common one is the RGB color space that represents the primary colors (red, green, and blue) where other colors are combined from these primary colors [10]. The RGB color space consists of a three-dimensional triple where each cell in the 3d triple represents the severity of the color with an integer value between 0 and 255. Where the red color value in RGB is (255, 0, 0), the green color value is (0, 255, 0) and the blue color value is (0, 0, 255) [11].

3.1 Six-Dimensional Hyperchaotic System

The development in field communication technology and cryptography led to the development of a new system with complex comparative characteristics that involve inefficient image encryption. Hyperchaotic System is defined as “a chaotic system with more than one positive Lyapunov exponent” [12]. Lyapunov exponents (LE) are a “diagnostic tool for analyzing the presence of chaos in a system by providing a quantitative measure of the divergence or convergence of nearby trajectories, by averaging the expansion rate of the phase space” [13]. The Six-Dimensional Hyperchaotic System is defined by Equation (1) as follows:

$$\begin{cases} \dot{x}_1 = a(x_2 - x_1) + x_4 - x_5 - x_6 \\ \dot{x}_2 = cx_1 - x_2 - x_1x_3 \\ \dot{x}_3 = -bx_3 + x_1x_2 \\ \dot{x}_4 = dx_4 - x_2x_3 \\ \dot{x}_5 = ex_6 + x_3x_2 \\ \dot{x}_6 = rx_1 \end{cases} \quad (1)$$

Where the variables x_1, x_2, x_3, x_4, x_5 and x_6 are the state variables for Hyperchaotic System and the

variables a, b, c, d, e, and r represent a selected constant value and their values in order 10, 8/3, 28, -1, 8, and 3.

3.2 Fibonacci Q-matrix

The Fibonacci number is denoted by f_n in which each Fibonacci number is equal to the sum of the two preceding numbers [14]. Fibonacci number formula expressed by Equation (2) as follows:

$$f_n = f_{n-1} + f_{n-2} \text{ with } f_0 = 0 \text{ and } f_1 = 1 \quad (2)$$

The Fibonacci Q-matrix is a two-by-two square matrix defined as [15]:

$$Q = \begin{bmatrix} f_2 & f_1 \\ f_1 & f_0 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \quad (3)$$

Where f_n is a Fibonacci number.

4. THE PROPOSED ENCRYPTION ALGORITHM

The proposed encryption algorithm is used to encrypt RGB Images by involving a 6D hyperchaotic system and Fibonacci Q-matrix in three main steps: (1) generating a fractional hyperchaotic matrix (2) image scrambling (3) image pixel diffusion. **Figure 2** represents the main steps for one round of the encryption algorithm to generate an encrypted image.

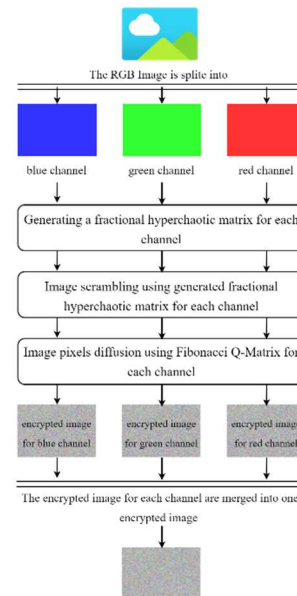


Figure 2: Flow Chart of the Proposed Encryption Algorithm.

For generating a fractional hyperchaotic matrix, we used the Runge-Kutta-Fehlberg method (RKF45) to get a numerical solution for a

hyperchaotic system of a fractional-order differential equation [16]. The values used in RKF45 are calculated by Equations 4 and 5 as follows:

$$\text{The initial value } (x_1) = \frac{\sum_{i=0}^{RC} f(i) + (R \times C)}{2^{23} + (R \times C)} \quad (4)$$

$$x_n = \text{modulo}(x_{n-1} \times 10^6, 1) \quad (5)$$

Where R and C represent the dimensions of the image (number of rows and columns) and $f(i)$ represent the flattening of the image matrix in column-major.

The sorted generated fractional hyperchaotic matrix obtained from the numerical solution of RKF45 is used to scramble the image by changing the position of an image pixel without changing its values. After that the diffusion is applied to the image pixels by multiplying each two-by-two sub-matrix from the scrambled image with the Fibonacci Q-Matrix by Equation 6 as follows:

$$\begin{bmatrix} D_{i,j} & D_{i,j+1} \\ D_{i+1,j} & D_{i+1,j+1} \end{bmatrix} = \begin{bmatrix} S_{i,j} & S_{i,j+1} \\ S_{i+1,j} & S_{i+1,j+1} \end{bmatrix} \begin{bmatrix} 89 & 55 \\ 55 & 34 \end{bmatrix} \text{ mod } 256 \quad (6)$$

Where D represents the Diffusion image and S represents the scrambled image.

The following pseudo-code explains the encryption algorithm in detail.

Algorithm 1 One Round of Encryption Algorithm

Inputs:

- 1: The RGB image M with dimensions $R \times C$.
 - 2: The constant variables for Hyperchaotic System are $a=10, b=8/3, c=28, d=-1, e=8$ and $r=3$.
 - 3: Hyperchaotic System equation interval from $T_{start} = .9865 \times (R \times C/3)$ to $T_{end} = \text{ceil}(R \times C/3)$
-

Steps:

- 1: Extract the blue, green, and red channels from M each has the shape of $R \times C$.
- 2: Repeat these steps for the three channels:
 - 2.1: Flatten the channel matrix into 1D matrix P by column-major.
 - 2.2: Calculate the initial condition x_1, x_2, x_3, x_4, x_5 and x_6 for state variables by the equations (4) and (5).
 - 2.3: Solve the Six-Dimensional Hyperchaotic System equation (1) by the calculated initial conditions by interval T_{start} and T_{end} to get the solution matrix Y.
 - 2.4: Discard the solution from the interval T_{start} in Y.

2.5: Reverse the matrix Y by changing the row elements into column elements and the column elements into row elements to get the Y^T .

2.6: Select columns 1,3 and 5 form Y^T to get Y^T .

2.7: Flatten the Y^T matrix into 1D matrix R by column-major.

2.8: Perform indirect ascending order sorting on R to get the matrix of indices I which would sort the matrix R.

2.9: Store the I matrix which represents the key for decryption.

2.10: Change matrix P values depend on the I matrix to get S.

2.11: Reshape matrix P to 2D Matrix by $R \times C$ to get P^r

2.12: Iterate over P^r matrix by i, j with a step of 2 to calculate the diffusion matrix D which represents the encrypted image by the equation (6).

3: Merge the three encrypted images into one image C.

4: **End**

outputs:

- 1: The encrypted image C.
 - 2: Three keys for each encrypted channel.
-

5. THE PROPOSED DECRYPTION ALGORITHM

The decryption algorithm aims to get the original image by encrypting RGB Image by using the generated keys for each channel in the encryption phase and Fibon-nacci Q-matrix by applying image pixels diffusion using Fibonacci Q-matrix de-fined in equation 7.

$$\begin{bmatrix} D_{i,j} & D_{i,j+1} \\ D_{i+1,j} & D_{i+1,j+1} \end{bmatrix} = \begin{bmatrix} S_{i,j} & S_{i,j+1} \\ S_{i+1,j} & S_{i+1,j+1} \end{bmatrix} \begin{bmatrix} 34 & -55 \\ -55 & 89 \end{bmatrix} \text{ mod } 256 \quad (7)$$

After that the image scrambling using the generated keys for each channel in the encryption phase. Figure 3 represents the main steps for one round of the decryption algorithm to restore the original encrypted image.

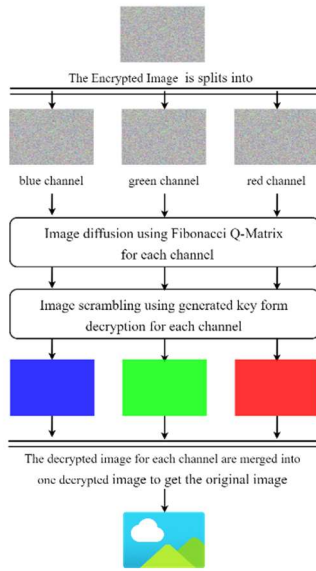


Figure 3: Flow Chart of the Proposed Decryption Algorithm.

The details of the proposed decryption algorithm are explained in the following pseudo-code.

Algorithm 2 One Round of Decryption Algorithm

Inputs:

- 1: The RGB image M with dimensions $R \times C$.
- 2: The keys for each channel I from the encryption phase.

Steps:

- 1: Extract the blue, green, and red channels from M each has the shape of $R \times C$.
- 2: Repeat these steps for the three channels:
 - 2.1: Iterate over the channel matrix by i, j with a step of 2 to calculate the diffusion matrix D which represents the encrypted image by the equation (7).
 - 2.2: Perform indirect ascending order sorting on I To get the matrix of indices P which would sort the matrix I .
 - 2.3: Flatten the D matrix into 1D matrix W by column-major.
 - 2.4: Change matrix W values depend on the P indices matrix to get C .
 - 2.5: Reshape matrix C to 2D Matrix by $R \times C$ to get C^r .
- 3: Merge the three C^r decrypted image into one image O .

4: End










outputs:

- 1: The original image O .

6. EXPERIMENTS AND RESULTS

The proposed algorithm was implemented using python language which is a high-level programming language, created by Guido van Rossum to emphasize code readability with the use of significant indentation [17]. Also, the proposed en-ryption/decryption algorithm was tested using different standard dimensions of RGB images as shown in Table 1.

Table 1: Sample RGB images.

Image name	Image preview	Image dimensions (pixels)
Sample_1		200 x 200
Sample_2		320 x 240
Sample_3		400 x 200
Sample_4		512 x 512
Sample_5		720 x 576
Sample_6		1024 x 768
Sample_7		1280 x 720
Sample_8		1280 x 1024
Sample_9		1920 x 1080



$$r = \frac{\sum_i(x_i-x_m)(y_i-y_m)}{\sqrt{\sum_i(x_i-x_m)^2} \sqrt{\sum_i(y_i-y_m)^2}} \tag{9}$$

6.1 Entropy Analysis

Entropy is a quantitative measure of image information content, used for image comparison. A good encryption algorithm produces an encrypted image that has an entropy as close as possible to the entropy of the original image. Entropy for RGB image is calculated for each channel by Equation 8 as follows [18]:

$$\text{The entropy} = -\sum_{i=1}^T P(p_i) \log_2 P(p_i) \tag{8}$$

Where T is the total number of pixels in the image and p_i is the probability of pixel i calculated by dividing the total number of pixel values counted in the image over the total number of image pixels.

Table 2 shows that the values of entropy for each channel in the original images and the decrypted images are almost identical. Overall, the entropy values of the original and decrypted images in all three channels remain consistent, indicating that the encryption process maintains the randomness and information content of the images across the red, green, and blue channels.

Table 2: Entropy Calculation For Each Channel Of Original Images And Their Decrypted

Image name	Entropy of original image			Entropy of decrypted image		
	Red Channel	Green Channel	Blue Channel	Red Channel	Green Channel	Blue Channel
Sample 1	7.7268	7.6733	7.3959	7.7268	7.6733	7.3959
Sample 2	7.4779	7.3217	7.3154	7.4779	7.3217	7.3154
Sample 3	7.2545	7.1217	7.0560	7.2545	7.1217	7.0560
Sample 4	7.6996	7.6288	7.7010	7.6996	7.6288	7.7010
Sample 5	7.7812	7.7090	7.6805	7.7812	7.7090	7.6805
Sample 6	7.8858	7.7695	7.4945	7.8858	7.7695	7.4945
Sample 7	7.7877	7.6190	7.3627	7.7877	7.6190	7.3627
Sample 8	7.8210	7.8553	7.6869	7.8210	7.8553	7.6869
Sample 9	7.5032	7.3949	6.8872	7.5032	7.3949	6.8872
Sample 10	7.7526	7.7831	7.8352	7.7526	7.7831	7.8352

6.2 Coefficient of Correlation

Another approach to showing the efficiency of the proposed algorithm is to calculate the coefficient of correlations by measuring the similarity between the pixels in each channel of the original image with the pixels in each channel of the encrypted image [19]. correlation coefficient r is calculated by Equation 9 as follows:

Where, x_i, y_i represent the intensity values for the original image and encrypted image respectively and x_m, y_m represent the mean intensity values for the original image and encrypted image respectively.

The value of the coefficient of correlations and its meaning are as follows:

- 1, means the two images are identical.
- 0, means the two images are completely uncorrelated.
- 1, means the two images are completely anti-correlated.

A good algorithm should have a zero value for the correlation coefficient. Figure 4 shows that the correlation coefficient is nearest zero from the image Sample_1 and its encrypted image. Where O_R, O_G and O_B represent the channel for the Sample_1 image and E_R, E_G and E_B represent the channels for the encrypted image.

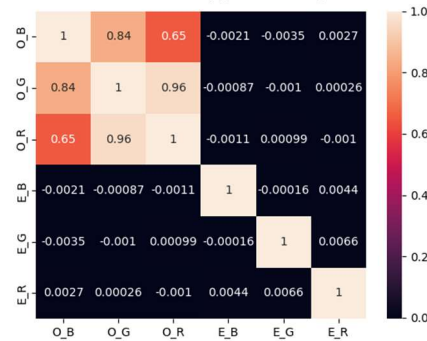










Figure 4: Correlation Coefficient Between Each Channel In The Original Image And Encrypted Image

6.3 The proposed algorithm against Image noise

Images are affected by a different type of noise that may change their clarity, pixels and properties, and as a result, degrade their quality. Image noise is defined as “s a random variation of image intensity and visible as a part of grains in the image” [10]. To measure the availability in our proposed algorithm, we add different types of noise that the encrypted image may be exposed during transmission over the network [20]. The result shows that image contents are still recognized which approved the availability of the proposed algorithm against noise as shown in Table 3.

Table 3: Test Result For The Proposed Algorithm Against Noise

Noise type	Original image	Decrypted image
salt-and-pepper noise(amount = 0.002)		
salt-and-pepper noise(amount = 0.005)		
Gaussian noise(Mean of distribution=0.1 And Deviation of distribution=0.1)		
Gaussian noise (Mean of distribution=1 And Deviation of distribution=0.2)		

Through extensive simulations and experimental testing as shown in Table 3, we have quantitatively assessed the algorithm's performance in the presence of different levels and types of image noise. The results indicate that our algorithm not only maintains robust encryption under noisy conditions but also outperforms existing methods in terms of noise tolerance and image quality preservation. Furthermore, practical applications of the proposed algorithm in real-world scenarios have demonstrated its effectiveness in securing image data against noise-induced vulnerabilities. By providing a reliable defense mechanism against image noise, our algorithm offers enhanced protection for critical image assets in various domains, including medical imaging, surveillance, and multimedia communication.

Overall, the results obtained from our evaluation highlight the efficacy of the proposed algorithm in safeguarding image integrity and confidentiality in noisy environments. The robustness demonstrated against image noise underscores the practical utility and reliability of our encryption solution in ensuring data security in the face of potential noise disturbances.

7. ENHANCE PROPOSED ALGORITHM PERFORMANCE

An analysis has been performed on the implemented algorithm performance to identify performance bottlenecks by using a built-in module in python called "cProfile" [26]. Table 4 presents data on the CPU time taken for executing encryption and decryption tasks on a single core for various

image sizes. The encryption time increases as the image dimensions grow, reflecting the larger data volume that needs processing. For instance, encryption time ranges from 1293.1 ms for Sample_1 to 577141.5 ms for Sample_10. The time taken to generate the fractional hyperchaotic matrix, a crucial part of the encryption process, is a significant portion of the encryption time, as indicated by percentages relative to the total encryption time.

This generation time also escalates with larger image sizes. Decryption time, though generally shorter than encryption time, also rises with larger image sizes as it involves recovering the original image from the encrypted data. The percentage of encryption time spent on generating the fractional hyperchaotic matrix ranges from approximately 80% to 82.6%, highlighting its substantial role in the encryption process. Overall, the analysis underscores how image size impacts CPU processing times for encryption and decryption tasks, with the generation of the fractional hyperchaotic matrix playing a notable role in the encryption process.

The computer hardware specification used for implementation and experiment testing is as follows:

- ✓ Processor name: AMD Ryzen 9 5900HX.
- ✓ The number of cores in the Processor: 8.
- ✓ The number of threads: 16.
- ✓ Processor Core Speed: 3606.5 MHz
- ✓ Memory type and size: DDR4 (32 GB).

Table 4: CPU time elapsed for execution encryption and decryption on one core.

Image name	Encryption time elapsed	generating fractional hyperchaotic matrix time elapsed	Decryption time elapsed
Sample_1	1293.1 ms	1066.8 ms (82.5% of Encryption time elapsed)	221.8 ms
Sample_2	2427.9 ms	1957.7 ms (80.6% of Encryption time elapsed)	416 ms

Sample_3	2512.5 ms	2060.8 ms (82% of Encryption time elapsed)	447.5 ms
Sample_4	7993.5 ms	6603.5 ms (82.6% of Encryption time elapsed)	1241.6 ms
Sample_5	12606.5 ms	10368 ms (82.2% of Encryption time elapsed)	2057.2 ms
Sample_6	24077.2 ms	19554.2 ms (81.2% of Encryption time elapsed)	3953.5 ms
Sample_7	25827.3 ms	21289.3 ms (82.4% of Encryption time elapsed)	4451.6 ms
Sample_8	36894 ms	30391.9 ms (82.4% of Encryption time elapsed)	6328.5 ms
Sample_9	59233 ms	48742.5 ms (82.3% of Encryption time elapsed)	10320 ms
Sample_10	577141.5 ms	471426.3 ms (81.7% of Encryption time elapsed)	105636.8 ms

7.1 Parallelizing Encryption Algorithm

Parallel and multiprocessing methods split large numerical problems into smaller subproblems, reducing total computation time by taking advantage of multiple cores on one machine. In general, a program running on N cores is N times faster than running the same program on one core [27]. Parallelism is achieved in python by creating multiple processes with their own separate global interpreter lock (GIL) to be executed on multiple

cores. GIL is a mutex that pre-vents multiple threads from executing Python bytecodes at once. Since each pro-cess has its own GIL, they will run in parallel at the same time [28].

Python has a built-in module for multiprocessing that provide an API to take advantage of mul-tiprocessing with two main classes which are the process class and the pool class. Both classes use the FIFO policy to schedule execution over processes. However, in Pool Class memory only contain the process that is executing. But In the Pro-cess class, the memory contains all processes to be executed. The picking meth-odology depends on the number of jobs, IO activity and the amount of data (im-age dimensions in our case).

The pool class allow execution function across mul-tiple input arguments by distributing the tasks over the machine cores. Our pro-posed encryption algorithm consists of repeating the main steps on each channel of the RGB image sequentially. we can perform encryption tasks for the three channels in parallel (multi-core) as shown in Figure 5.

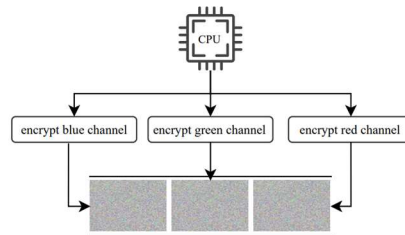


Figure 5: Parallel processing for the encryption algorithm

For our situation, we re-implement our proposed encryption algorithm using Pool Class and make a performance analysis using the cProfile module. Table 5 summarizes the result.

Table 5: Performance analysis summary

Image name	Number of cores	Encryption time elapsed	speedup
Sample_1	One core	1293.1 ms	
	Two cores	2612.6 ms	
	Three cores	2903.7 ms	
	Four cores	2916.2 ms	
Sample_2	One core	2427.9 ms	
	Two cores	3257.7 ms	

	Three cores	3304.4 ms	
	Four cores	3413.9 ms	
Sample_3	One core	2512.5 ms	
	Two cores	3356.1 ms	
	Three cores	3369.5 ms	
	Four cores	3383.3 ms	
Sample_4	One core	7993.5 ms	
	Two cores	6582.4 ms	1.21
	Three cores	5227.3 ms	1.52
	Four cores	5205.3 ms	1.54
Sample_5	One core	12606.5 ms	
	Two cores	9261.9 ms	1.36
	Three cores	6806.1 ms	1.85
	Four cores	6697.2 ms	1.88
Sample_6	One core	24077.2 ms	
	Two cores	15696.6 ms	1.53
	Three cores	10478.1 ms	2.30
	Four cores	10287.6 ms	2.34
Sample_7	One core	25827.3 ms	
	Two cores	18195 ms	1.41
	Three cores	12135.1 ms	2.13
	Four cores	11560.4 ms	2.23
Sample_8	One core	36894 ms	
	Two cores	28225.4 ms	1.30
	Three cores	15252.9 ms	2.42
	Four cores	15075.4 ms	2.45
Sample_9	One core	59233 ms	
	Two cores	44296.5 ms	1.34

	Three cores	22834.9 ms	2.59
	Four cores	22121.4 ms	2.68
Sample_10	One core	598352.4 ms	
	Two cores	434988.5 ms	1.37
	Three cores	216914.6 ms	2.76
	Four cores	212517.3 ms	2.82

The execution time of encryption was decreased while increasing the number of cores, the minimum speedup we achieved was 1.30 and the maximum speedup was 2.82 since all encryption internal operation is “CPU-bound”. For a small image’s dimensions, the execution time on one core is less than the execution time on 2, 3 and 4 cores because of the overhead of creating a Pool and process management. Overall, the results show that employee parallelization has improved the algorithm's performance.

8. COMPARISON OF THE PROPOSED ALGORITHM WITH OTHER ALGORITHMS

A common standred image called “Lena” with dimensions 512x512 shown in Table 6 is used to compare the proposed algorithm with other algorithms based on entropy values for red, green and blue channels.

Table 6: Lena image and entropy values for its channels.


Image	Entropy for Red Channel	Entropy for Green Channel	Entropy for Blue Channel
	7.9974	7.9971	7.9973

Table 7 provided with the entropy values for the red, green, and blue channels of Lena image encrypted using different algorithms. The proposed algorithm maintains relatively high entropy values across all three channels, indicating a good level of randomness and information content preservation during encryption. it can be highlighted that the proposed algorithm stands out as superior compared to the other algorithms in terms of entropy

preservation and randomness in the encrypted image.

Here are the key points supporting the superiority of the proposed algorithm:

- The proposed algorithm consistently maintains high entropy values across all three channels (Red: 7.9974, Green: 7.9971, Blue: 7.9973), indicating a high level of randomness and information content preservation.
- When compared to the entropy values of the other algorithms, the proposed algorithm generally exhibits higher entropy values, suggesting better encryption quality in terms of information security and randomness.
- The high entropy values across all channels in the proposed algorithm imply a robust encryption process that effectively preserves the information content of the Lena image.

Therefore, based on the entropy analysis provided, it can be concluded that the proposed algorithm demonstrates superior performance in encrypting the Lena image compared to the other algorithms listed in the table.

Table 7: The comparison result of the proposed algorithm with different other algorithms

Ref	Entropy for Red Channel	Entropy for Green Channel	Entropy for Blue Channel
The proposed algorithm	7.9974	7.9971	7.9973
[3]	7.9973	7.9969	7.9971
[4]	7.9991	7.9973	7.9967
[5]	7.9966	7.9972	7.9967
[9]	7.9917	7.9912	7.9918

Our RGB image encryption algorithm builds upon the foundations laid by previous works, offering innovative enhancements and advancements in the field of image security. Firstly, while [3] focused on grey image encryption utilizing a 6D hyperchaotic system with double confusion and diffusion techniques, our algorithm extends this methodology to RGB images by harnessing a 6D fractional-order hyperchaotic Chen system. This sophisticated approach enables enhanced scrambling of each color channel, providing a more comprehensive encryption process tailored to the unique characteristics of colored images.

In contrast to the color image encryption algorithm proposed in [4], which relied on a 4D

hyperchaotic Chen system, our algorithm introduces a groundbreaking element - the Fibonacci Q-matrix for image diffusion. This strategic addition elevates the level of security and complexity in our encryption process, aiming to instill a higher degree of chaotic behavior and fortify resilience against potential attacks. By integrating the Fibonacci Q-matrix, we transcend the limitations associated with lower-dimensional systems, paving the way for enhanced encryption robustness.

Moreover, unlike the encryption approach combining a 3D hyperchaotic Chen system and stack data structure as seen in [5], our method incorporates multiprocessing to elevate algorithm performance to new heights. Leveraging parallelism as a key strategy, our algorithm is designed to optimize the encryption process, streamline execution time, and enhance scalability. This integration addresses critical concerns surrounding processing overhead and algorithmic efficiency highlighted in contemporary literature, positioning our approach at the forefront of efficient and effective image encryption techniques.

In summary, our RGB image encryption algorithm represents a significant leap forward in the realm of image security. By synergizing elements of hyperchaotic systems, Fibonacci matrices, and multiprocessing techniques, we have crafted a solution that ensures robust security measures, efficient encryption processes, and seamless compatibility with modern image encryption standards.

9. CONCLUSION

A new image encryption algorithm using a 6D hyperchaotic system and Fibonacci Q-matrix is presented in this paper. Confusion property is achieved by generating a fractional hyperchaotic matrix and diffusion through the Fibonacci Q-Matrix, respectively. The security of the proposed algorithm is analyzed via several security tests. The result shows that the proposed algorithm was able to overcome the different types of noise and attacks. The proposed algorithm surpassed the other algorithms mentioned in section 6 in terms of entropy and how the original image was retrieved through the key without change or loss of image pixels. In this paper, we have employed parallelism to improve performance and we reduced the execution time for the encryption stage. The presented image encryption algorithm utilizing a 6D

hyperchaotic system and Fibonacci Q-matrix demonstrates effectiveness in overcoming various noise and attacks, a limitation lies in the need for further evaluation of its robustness against advanced cryptanalysis techniques, such as differential and linear cryptanalysis. Future research directions include exploring the algorithm's performance under a broader range of attack scenarios to assess its real-world security resilience, scalability to larger image sizes, and datasets, as well as investigating enhancements through additional cryptographic primitives or hybrid encryption schemes. Moreover, user studies or surveys could provide valuable insights into usability and user experience aspects, guiding refinements for optimizing the encryption process and ensuring practical applicability in diverse image encryption scenarios.

REFERENCES:

- [1] Radke, Shashikant & Mishra, Dharendra, "Review of Image Security approaches: Concepts, Issues, Challenges and Applications," *Journal of University of Shanghai for Science and Technology*, vol. 23, pp. 1047-1054, 2021.
- [2] Qayyum, Abdullah & Ahmad, Jawad & Boulila, Wadii & Rubaiee, Saeed & Arshad, Arshad & Masood, Fawad & Khan, Fawad & Buchanan, William, "Chaos-Based Confusion and Diffusion of Image Pixels Using Dynamic Substitution," 2020.
- [3] Hosny, Khalid & Kamal, Sara & Darwish, Mohamed & Papakostas, George, "New Image Encryption Algorithm Using Hyperchaotic System and Fibonacci Q-Matrix," *Electronics*, vol. 10, 2021.
- [4] Hosny, Khalid & Kamal, Sara & Darwish, Mohamed, "Novel encryption for color images using fractional-order hyperchaotic system," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, 2022.
- [5] Khodadadi, H & Mirzaei, Omid, "A stack-based chaotic algorithm for encryption of colored images," *Journal of AI and Data Mining*, vol. 5, pp. 29-37, 2017.
- [6] Liu, L., Zhang, Q. & Wei, X, "A RGB image encryption algorithm based on DNA encoding and chaos map," *Computers and Electrical Engineering*, vol. 38, pp. 1240-1248, 2012.
- [7] Wei, X., Guo, L., Zhang, Q., Zhang, J. & Lian, S, "A novel color image encryption algorithm based on DNA sequence operation and hyper-chaotic system," *The Journal of Systems and Software*, vol. 85, pp. 290-299, 2012.
- [8] Murillo-Escobar, M. A., Cruz-Hernández, C., Abundiz-Pérez, F., López-Gutiérrez, R. M. & AcostaDel Campo, O.R, "A RGB image encryption algorithm based on total plain image characteristics and chaos," *Signal Processing*, vol. 109, pp. 119-131., 2015.
- [9] Aljazeera, Ibtisam & Alrikabi, Haider & Alaidi, Abdul Hadi, "Encryption of Color Image Based on DNA Strand and Exponential Factor," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 18, pp. 101-113, 2022.
- [10] Al Azzeh, Jamil & Zahran, Bilal & Alqadi, Ziad, "Salt and Pepper Noise: Effects and Removal," *International Journal on Informatics Visualization*, 2018.
- [11] Khrisat, Mohmmad & Alqadi, Ziad & Dwairi, Majed & hindi, amjad & Eltous, Yousif & Khawatreh, Saleh, "Analysis of Color Image Noising Process," vol. 9, pp. 65-72, 2020.
- [12] Yuan Gao, Chenghua Liang, "A New 4D Hyperchaotic System and Its Generalized Function Projective Synchronization," *Mathematical Problems in Engineering*, p. 13, 2013.
- [13] Juan C. Vallejo, Miguel A F Sanjuán, "Lyapunov Exponents," in *Predictability of Chaotic Dynamics*, 2017.
- [14] T. Koshy, "Fibonacci and Lucas Numbers with Applications," 2001.
- [15] U. Pagalay, "Numerical solution for immunology tuberculosis model using Runge Kutta Fehlberg and Adams Bashforth Moulton method," *Jurnal Teknologi*, vol. 78, no. 5, pp. 369-372, 2016.
- [16] D. Kuhlman, *Beginning Python, Advanced Python, and Python Exercises*, 2012.
- [17] Tsai, D. Y., Lee, Y., & Matsuyama, E, "Information entropy measure for evaluation of image quality," *Journal of digital imaging*, vol. 21, no. 3, p. 338-347, 2008.
- [18] Avneet Kaur, Lakhwinder Kaur, Savita Gupta, "Image Recognition using Coefficient of Correlation and Structural SIMilarity Index in Uncontrolled Environment," *International Journal of Computer Applications*, pp. 32-39, 2012.
- [19] Win, Ni & Kyaw, Khin & Win, Thu & Aung, Phyo, "Image Noise Reduction Using Linear and Nonlinear Filtering Techniques," *International Journal of Scientific and Research Publications (IJSRP)*, vol. 9, 2019.

- [20] Chai, X., Fu, X., Gan, Z., Lu, Y., & Chen, Y, "A color image cryptosystem based on dynamic DNA encryption and chaos," *Signal Processing*, vol. 155, pp. 44-62, 2019.
- [21] Pak, C., An, K., Jang, P., Kim, J., & Kim, S, "A novel bit-level color image encryption using improved 1D chaotic map," *Multimedia Tools and Applications*, vol. 78, no. 9, pp. 12027-12042, 2019.
- [22] ur Rehman, A., Liao, X., Ashraf, R., Ullah, S., & Wang, H, "A color image encryption technique using exclusive-OR with DNA complementary rules based on chaos theory and SHA-2," *Optik*, vol. 159, pp. 348-367, 2018.
- [23] Wu, X., Wang, K., Wang, X., Kan, H., & Kurths, J, "Color image DNA encryption using NCA map-based CML and one-time keys," *Signal Processing*, vol. 148, pp. 272-287, 2018.
- [24] Zhang, Y. Q., He, Y., Li, P., & Wang, X. Y, "A new color image encryption scheme based on 2DNLCML system and genetic operations," *Optics and Lasers in Engineering*, vol. 128, p. 106040, 2020.
- [25] "The Python Profilers," Python Software Foundation, [Online]. Available: <https://docs.python.org/3/library/profile.html>. [Accessed may 2022].
- [26] S. Y. Yu, S. R. Chhetri, A. Canedo, P. Goyal, and M. A. Al, "Pykg2vec: A python library for knowledge graph," *arXiv*, vol. 22, pp. 1-6, 2019.
- [27] "GlobalInterpreterLock," 2020. [Online]. Available: <https://wiki.python.org/moin/GlobalInterpreterLock>. [Accessed 15 June 2022].