Indian Journal of Engineering & Materials Sciences
Vol. 31, October 2024, pp. 687-698
DOI: 10.56042/ijems.v31i5.11330

Performance analysis of modified rubik's cube algorithm for secured image security applications

S. Sharmila[*a], R. S. Bhuvaneswaran[a] & Dhandapani Vaithiyanathan[b]

[a]Ramanujan Computing Centre, College of Engineering Guindy, Anna University, Chennai 600 025, India

[b]Departement of Electronics and Communication Engineering, National Institute of Technology, Delhi 110 036, India

Received: 23 May 2024; accepted: 10 July 2024

In recent years, several encryption methods based on various encryption algorithms available in newer systems have been presented to safeguard digital photographs from attackers. This work provides a modified Rubik's cube image encryption technique based on the Rubik's cube principle. In the proposed framework, the modified Rubik's cube algorithm repeatedly rotates each picture face to encrypt it. The proposed updated Rubik's cube algorithm consists of the following steps: Load the images: Load the image to encrypt, then convert to RGB; if the image is not already in RGB format, convert it. Create a Rubik's Cube: Represent the Rubik's cube by creating a random key, either as a 3D array or with a custom data structure, and then scrambling the cube to serve as the encryption mechanism. Use Encryption: Scramble the image data with the Rubik's cube algorithm. We used many iterative modulo 2 functions with left or right circular shifting operations to encrypt each row and column. Implement decryption by reversing the scrambling process. In the decryption for row and column, we used numerous iterative modulo 2 functions with left or right circular shifting. The reported and proposed algorithms are tested for robustness and strength using MATLAB tools.

Keywords: Encryption, Decryption, Cryptography, Scrambling, Rubik's cube, Structural similarity index measure (SSIM)

1 Introduction

In recent years, information sharing through multimedia environments has increased rapidly across the globe. Protecting private information from hackers and unauthorised access is one of the biggest problems of the modern world. Many Government and private organizations deal with confidential and sensitive data such as text documents, images, etc. For the last three decades, there are several algorithms were proposed and reported by the research community to secure the data. At the same time, the attacker and unauthorized agencies keep on attempting to seize data. So, there is a need to modify the encryption algorithm or find a novel algorithm to ensure data security. The major objectives of the encryption techniques are to ensure high security from hackers and also to have very fast encryption and decryption operation. The primary use of encryption is to make sure that the data is well safeguarded will not be tampered unauthorized persons/agencies and can be decrypted only by the authorized receiver to recover the original information/data.

Encryption employs two methods steam ciphers and block ciphers. While block ciphers use static-length bits known as blocks to encrypt, stream ciphers encode one bit at a time[1]. Further cryptographic encryption procedures are split into two categories: symmetric and asymmetric encryption. To safeguard data from hackers, many academics have released a range of new or modified encryption and decryption approaches. Nair *et al.*[2] presented three phases of the encryption process, including Rubik's cube scrambling, bitmap shambling, and frame spin for colour pictures. Because of the duration of the scrambling sequence and rising block sizes, the author of the provided study concludes that the first step is tough[2]. As a result, the key's size can be adjusted to match the level of security required for certain applications or purposes. Kumari *et al.*[3] provide a quick overview of several picture encryption approaches. In their study, the authors examined and presented five chaos-based and ten traditional encryption algorithms. They experimented with three photos of different sizes such as 128x128, 192x192, and 256x256, and various performance metrics. Furthermore, based on the experimental study, the author discussed increased information security as well as temporal complexity. The user selects an algorithm based on security and temporal complexity, as well as requirements such as lower power

*Corresponding author (E-mail: sharme.sharmila@gmail.com)

consumption or faster operations. Monika and Pradeep presented a relative assessment of symmetrical key encoding algorithms[4], covering several symmetric encryption techniques, including Rivest Cipher 4 (RC4), Rivest Cipher 6 (RC6), Data Encryption Standard (DES), Triple Data Encryption Algorithm (TDES), Advanced Encryption Standard (AES), and Blowfish. The authors[4] found that the symmetric algorithm is faster than the asymmetric approach and uses less memory space, while the Blowfish encryption algorithm outperforms the other symmetric algorithms.

Many scholars have recently presented picture cryptography, which employs a Rubik's cube-based scrambling technique. Govinda and Prasanna investigate how the Rubik's cube and the game of life might be used to develop a generic picture encryption that provides improved security and efficiency[5]. In[6], the augmented Rubik's cube encryption approach developed in Java was evaluated and compared to the traditional Rubik's cube principle. Additionally, the server's performance was examined. Raniprima *et al.*[7] published an article on image steganography using the Rubik's cube encryption technique. Visual assault, Brute-Force attack, histogram analysis, arithmetical attack of chi-square examination, avalanche effect, and various picture sizes such as 32x32, 64x64, and 128×128 as well as embedding position are utilized to verify the efficiency of the presented system[7]. The proposed technique in[8] aims for image encryption or decryption using a customized pixel scrambling algorithm. The creator generates the keys using a random Rubik's cube arrangement, resulting in almost 43 quintillion key combinations[8]. Loukhaoukha *et al.*[9] recommended an encryption method for iris pictures based on permutation and diffusion. This improves the system's security against replay assaults. In this method[9], The original input image is first separated into blocks and permuted with a permutation key. Furthermore, each block's picture is scrambled using the Rubik's cube approach. To adjust the pixel values in the scrambled image, the XOR operator is applied separately to columns and rows. The author evaluated the algorithm on four iris photos and claimed that it is substantially more resistant to typical attacks while also achieving a high level of security[9]. Zhong and Li's[10] multi-image encryption solutions utilized three-dimensional (3D) shuffling scrambling as well as the Haar wavelet transform. The author of this paper created a dynamic pseudo-random sequence generator based on a chaotic library that contains three different types of one-dimensional (1D) chaotic maps. These maps are quite effective and have a good level of complexity, and they are closely related to images with plain text. To encrypt several photos, they must first be split into blocks, sliced, and then pieced back together to form a three-dimensional cube image[10]. Every layer of the image cube undergoes the Haar wavelet treatment, and the three-dimensional shuffle algorithm jumbles the low-frequency components. The cube was then rebuilt using the jumbled high-frequency components and low-frequency coefficients. Ultimately, following the reconstruction, each layer of the picture cube is subjected to a bitwise XOR operation using the chaotic matrix. Any size of colour or grayscale image can be encrypted using the suggested algorithm in[10]. In addition, the encryption technique is faster, more adaptable, and capable of safeguarding the attacker's data simultaneously. A recursive approach for scrambling images based on the Rubik's Cube is obtainable in[11]. A pseudo-random number producer is used to construct a long sequence that will act as the secret key in this work. This approach first splits the image in half and then joins the pixels from each side to form a cube, which is then jumbled using moves generated by a pseudo-random number generator. In addition, the recursive approach was applied to jumble the produced halves. The scrambling algorithm must be adaptable enough to couple with any specific confusion, diffusion, or substitution technique to provide complicated and well-secured encryption[11]. Based on a one-time pad, Ge *et al.*[12] suggest edanimageen coding method for information concealing. This approach[12]uses a hyperchaotic system to create a random parameter together with value transformation and position scrambling structures. Divide and conquer and similar key attacks will be thwarted, and the visual effects won't be affected by the decryption error brought on by the random parameter's embedding. The logistic chaotic order and the standard Rubik's cube are combined by the author in[13] to offer the scrambling algorithm. Similar to all other Rubik-based encryption techniques, the authors of this study[13]create cubes by dividing the original image into several blocks. To create the logistic system, rotate these cubes similarly to how you would a Rubik's cube, due to the Rubik's cube's intricacy and the revolution methods shaped by the chaotic scheme. The logistic system's arguments

alone might reverse this algorithm, allowing for one-key, one-image processing and utilizing the influences as the key.

The RC6 procedure and Rubik's cube are the foundation of Helmy *et al.*[14] proposed 3D image encryption technique, which starts with RC6 and encrypts numerous images independently. The 3D Rubik's cube is then used to encrypt further the received encrypted images. As a result, dispersal and variation in the encoded images are assured by the two-step encryption[14]. Gandhi *et al.*[15] described encrypting text, grayscale, and colour images with the Rubik's cube. Using the Rubik's cube as a guide, they jumble the original image by simply rearranging the pixel positions. The XNOR procedure is then performed on the even columns and rows using two separate secret keys. This process is then repeated for the odd columns and rows. The secret keys in this case are manually selectable and are generated at random. The author tested the encryption algorithm's durability against a variety of assaults, including statistical and differential attacks[15]. Using Rubik's cube technology, Mudduluri *et al.*[16] demonstrated sophisticated image encryption and decoding in 2022. With this method, two randomly generated vectors are utilized to generate the keys, and the same vector can be used for bitwise operations. Several types of attacks, including parametric and analytical attacks, are taken into consideration to analyze the algorithm[16]. According to the Rubik-based encryption technique in[17], it has a shorter processing time, requires less power, and is more suited for handheld devices such as mobile phones, among other reasons. Furthermore, the author claimed that the speed of the projected procedure is equivalent to that of the AES-128 encoding technique based on an experimental study[17]. According to[18], in recent years, various encryption approaches have emerged that leverage chaotic systems to secure digital images against cryptographic attacks. However, these encryption techniques often use small key areas, resulting in insufficient security. In work[18], the author presents a novel picture encryption method based on the Rubik's cube notion that can perform encryption and decryption quickly, which is critical for real-time applications. Abdull at if *et al.*[19] propose hybrid picture encryption methods using Rubik's cube and dynamic Deoxyribonucleic Acid (DNA) indoctrination methods. The Rubik's cube approach allows you to randomly arrange both the colour channels and the pixel placements in a picture. The pixel data are then encrypted using the dynamic DNA encoding approach[19]. Vidhya and Brindha developed a chaos-based picture encryption system that makes use of the Rubik's cube and the prime factorization process[20]. The authors[20] claimed that they achieved strong diffusion by using a novel prime factorization approach to create key orders for unassuming XOR-based dispersal with high sensitivity. A new chaotic picture encryption system is described in[21], This incorporates a block image scrambling system and a new dynamic index-based diffusion scheme. In this study, the dynamic index system is used to diffuse the jumbled image. In addition, for this work, we referred to one-dimensional chaotic systems for picture encryption[22] and explored applications and limitations in a survey on chaos-based image encryption[23].

In this research, we describe a modified Rubi's cube encryption technique that protects digital images from cryptographic assaults. If the image is not already in RGB format, it is first converted to it. The Rubik's cube idea is used, along with random key creation using a bespoke data structure or 3D array structure. Scramble the image data using the Rubik's cube algorithm. Furthermore, in this work, we use numerous iterative modulo 2 functions with left or right circular shifting to encrypt the row and column. These procedures can be repetitive until the amount of iterations is no longer touched. The reverse process will be used to decrypt and recover the original image. The new encryption algorithm is tested for security and validity by simulating and comparing its encryption and decryption techniques.

2 Materials and Methods

This work is based on the Rubick's cube algorithm in[18], in the suggested framework the modified Rubik's cube algorithm performs the identical rotation operation on each picture face many times to encrypt it. The proposed updated Rubik's cube algorithm involves the following steps:

a Load the Image: Load the image you want to encrypt.
b Convert to RGB: If the image is not in RGB format, convert it.
c Create Rubik's Cube: In MATLAB, represent the Rubik's cube as a 3D array or using a custom data structure.
d Scramble the Cube: Use the Rubik's cube to encrypt information.

e Encryption: Scramble the image data with the Rubik's cube algorithm.
In the encryption, for row and column, we have incorporated multiple iterative using modulo 2 functions with left or right circular shifting operation. More details of the algorithm are detailed in section 2.1

f Save or Display Encrypted Image: Save the encrypted image or display it.

g Decryption: Implement decryption by reversing the scrambling operation.

In the decryption, for row and column, we have incorporated multiple iterative using modulo 2 functions with left or right circular shifting operation. More details of the algorithm are detailed in section 2.2. Real-world encryption requires much more robust and secure cryptographic algorithms. Additionally, the encryption method presented here is very simplistic and provides strong security. It's more of a visual demonstration of how modified Rubik's cube movements can be converted into image encryption. The upcoming section describes the suggested encryption and decryption algorithms based on the Rubik's cube procedure. The block diagram of the modified Rubik's cube procedure is presented in Figs 1–2.

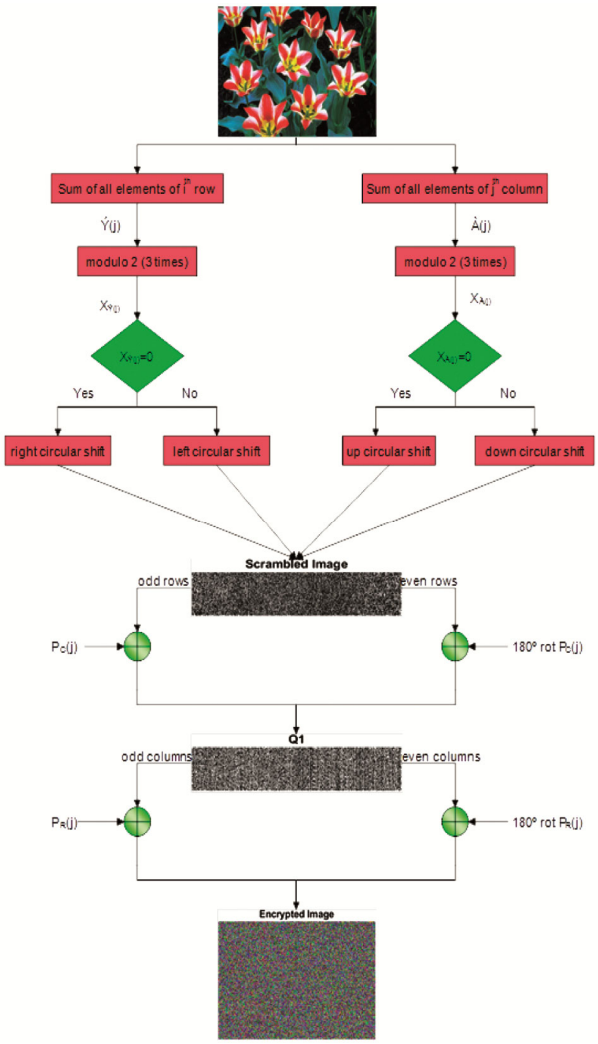

Fig. 1 — The framework of the suggested encryption method.

2. 1 A modified Rubik's Cube-based encryption algorithm

Let Q_0 be a γ-bit greyscale picture with dimensions X × Y. Q_0 denotes the picture element values matrix of image Q_0. The encryption algorithm involves the following steps:

(1) Create two random vectors (P_R and P_C) of lengths X and Y. Element P_R (i) and P_C (j). Each chooses a random value from the set W = {0,1,2,3, 4.....,2γ-1}. It is important to note that neither P_R nor P_C can have constant values.

(2) Set the amount of repetitions, ITR_{max}, and initialise the counter ITR to 0.

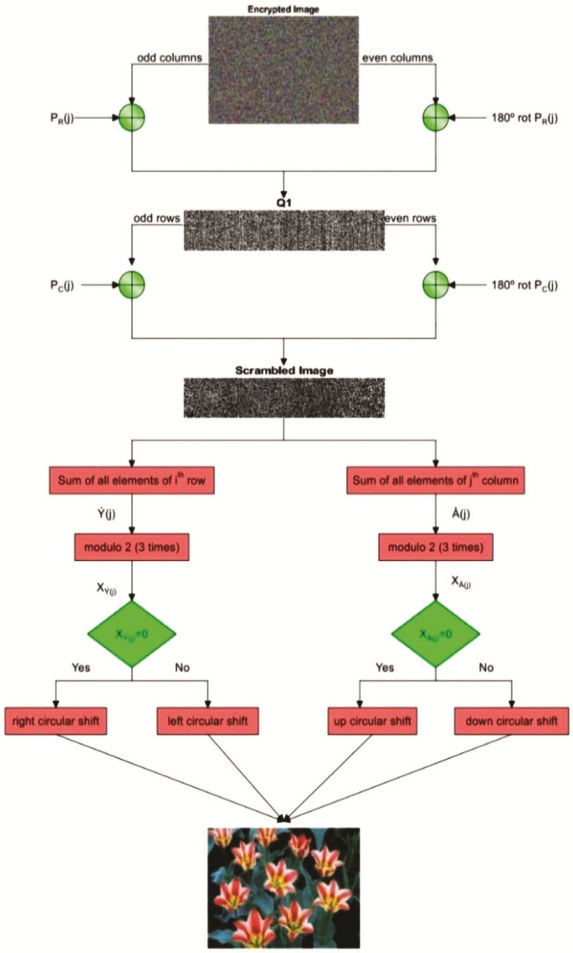

Fig. 2 — The framework of the suggested decryption method.

(3) Increase the counter by one: ITR = ITR+1.
(4) For each row i of picture Q0:
(a) Determine the total of all elements in row i, denoted by y(i).

$$y(i)=\sum_{j=1}^{Y} Q_0(i,j) i=1,2,3,4....., X. \quad \dots (1)$$

(b) Calculate modulo 2 of γ(i), represented by X_(γ(i)).
(c) Modulate X_(y(i)) by 2 to create a new set of X_(y(i)).
(d) Calculate modulo 2 of X_(y(i)), denoted by X_(y(i)), to get a new set of X_(y(i)).
(e) Row i is circularly shifted to the left or right using $P_R(i)$ positions, with the first pixel moving in the last pixel.

Compute the following step (e) for L - times to obtain the scrambled image from the original image to be encrypted, where L=1 to 3, L=1 to obtain the scrambled image and L=2 and 3 to re-scramble the scrambled image two times.

if $X_{\alpha(i)} = 0 \rightarrow$ make the right circular shift

else → make the left circular shift … (2)

(5) For each column j in image Q_0,
(a) Calculate the sum of all entries in column j, given as α(j).

$$\alpha(j) = \sum_{i=1}^{X} Q_0(i,j), \quad j=1,2,3,4,\ldots..,N \quad \ldots(3)$$

(b) Calculate modulo 2 of α(j), indicated by if X_α(j).
(c) Calculate modulo 2 of X_α(j) to create a new set of X_α(j).
(d) Again, calculate modulo 2 of X_α(j) to obtain a new set of X_α(j).
(e) Column j is circularly shifted down or up by $P_C(i)$ places as shown below:

Compute the following step (e) for L - times to obtain the scrambled image from the original image to be encrypted, where L=1 to 3, L=1 to obtain the scrambled image and L=2 and 3 to re-scramble the scrambled image two times.

if $X_{\alpha(j)}=0 \rightarrow$ make the up circular shift

else → make the down circular shift …(4)

Steps 4 and 5 above produce a scrambled image, denoted by Q_{SCB}.

(6) Using vector P_C, apply the bitwise XOR operation to each row of scrambled picture Q_{SCB} using the formulas below:

$$Q_1(2i-1, j) = Q_{SCB}(2i-1, j) \oplus P_C(j),$$

$$Q_1(2i, j)=Q_{SCB}(2i, j) \oplus \text{rot } 180(P_C(j)), \quad \dots (5)$$

The operators ⊕ and rot 180(P_C) express bitwise XOR and left-to-right vector flipping, respectively.

(7) Vector PR applies the bitwise XOR operator to each column of picture Q1 using the equations below:

$$Q_{ENC}(i, 2j-1) = Q_1(i, 2j-1) \oplus P_R(j),$$
$$Q_{ENC}(i, 2j) = Q_1(i, 2j) \oplus \text{rot } 180(P_R(j)), \quad \dots (6)$$

Rot 180(P_R) indicates a left-to-right flip of vector P_R.

(8) If ITR = ITRmax, the algorithm creates an encrypted picture I_{ENC} and ends the encryption process; otherwise, it moves on to step 3.

The secret keys in the proposed encryption algorithm are the vectors P_R, P_C, and the maximum iteration number, ITR_{max}. To achieve a fast encryption approach, set ITR_{max} to 1 (single repetition). In contrast, if $ITR_{max} > 1$, the approach becomes more secure since the key space is larger than when $ITR_{max} = 1$. ITR_{max} was confined to a single iteration.

2.2 Modified Rubik's Cube Decryption Algorithm

The decrypted picture, Q_0, is obtained using the encoded image, I_{ENC}, and the secret keys, P_R, P_C, and ITR_{max}, as given in the calculations below.

(1) Initialize ITR to 0.
(2) Increase the counter by one: ITR = ITR+1.
(3) Bitwise XOR is done on vector P_R and each column of encrypted picture I_{ENC}, as illustrated below.

$$Q_1(i,2j-1) = Q_{ENC}(i,2j-1) \oplus P_R(j),$$
$$Q_1(i,2j) = Q_{ENC}(i,2j) \oplus \text{rot } 180\ (P_R(j)), \quad \dots (7)$$

Using the P_C vector, apply the bitwise XOR operation to each row of image Q_1.

$$Q_{SCB}(2i-1, j) = Q_1(2i-1, j) \oplus P_C(j),$$
$$Q_{SCB}(2i, j) = Q_1(2i, j) \oplus \text{rot } 180(P_C(j)) \quad \dots (8)$$

(4) In each column j of the scrambled picture Q_{SCB},
(a) Find the sum of all elements in column j, denoted as $\alpha_{SCB}(j)$:

$$\alpha_{SCB}(j) = \sum_{i=1}^{X} Q_{SCB}(i, j), j = 1, 2, 3, 4,...,Y \quad \dots (9)$$

(b) Compute modulo 2 of $\alpha_{SCB}(j)$, represented by $X\alpha_{SCB}(j)$.
(c) Calculate modulo 2 of $X\alpha_{SCB}(j)$, represented as $X\alpha_{SCB}(j)$.
(d) Again, compute modulo 2 of $X\alpha_{SCB}(j)$, represented by $X\alpha_{SCB}(j)$.

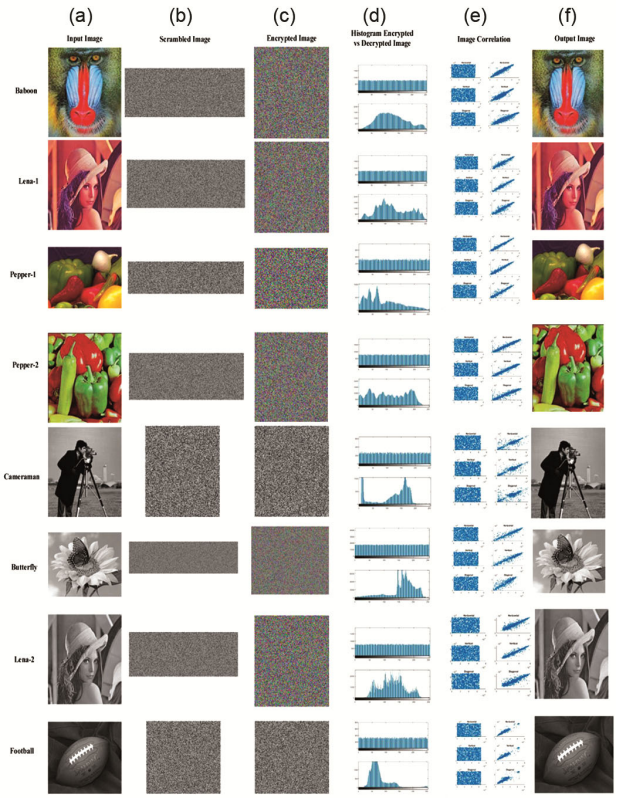

Fig. 3 — Rubik's Cube Image Encryption and Decryption.

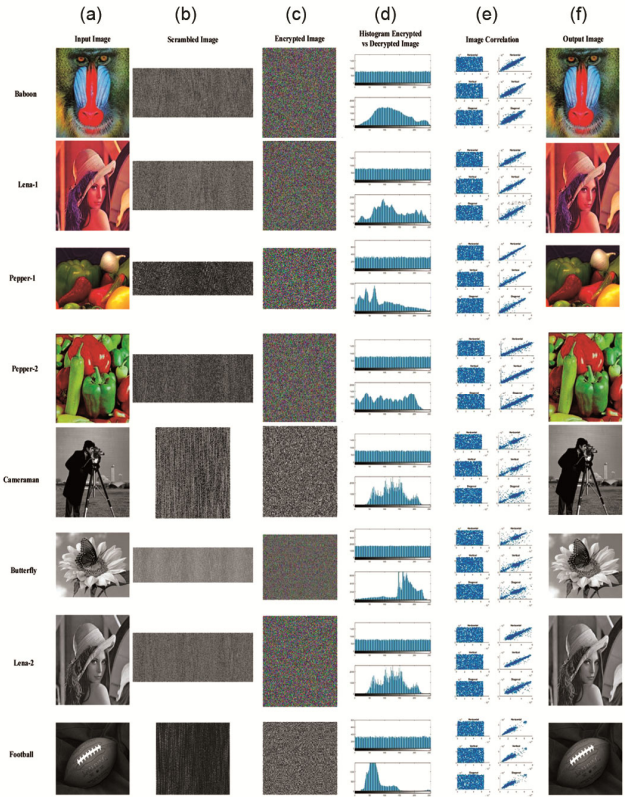

Fig. 4 — Modified Rubik's Cube Image Encryption and Decryption.

(e) Column j is circularly shifted down or up by $P_C(i)$ position as shown below: Compute the following step (e) for L - times to obtain the scrambled picture from the original image to be encrypted, where L=1 to 3, L=1 to obtain the scrambled image, and L=2 and 3 to re-scramble the scrambled image twice.

if $X\alpha_{SCB}(j)$, = 0 → up circular shift

else → make the down circular shift … (10)

(5) For each row i of scrambled image Q_{SCB},

(a) Calculate the total of all elements in row i, represented by $y_{SCB}(i)$:

$$\gamma_{SCB}(i) = \sum_{j=1}^{Y} Q_{SCB}(i, j), \quad i=1, 2, 3, 4.... X \quad \ldots (11)$$

(b) Calculate modulo 2 of $\gamma_{SCB}(j)$, represented by $X\gamma_{SCB}(j)$,

(c) Compute modulo 2 $X\gamma_{SCB}(j)$ denoted by $X\gamma_{SCB}(j)$, to obtain a new set of $X\gamma_{SCB}(j)$

(d) Again calculate modulo 2 of $X\gamma_{SCB}(j)$ represented by $X\gamma_{SCB}(j)$, to obtain a new set of $X\gamma_{SCB}(j)$

(e) $P_R(i)$ circularly shifts row i to the left or right, as shown below: $P_R(i)$ shifts row i in a circular motion to the left or right, as seen below:

Compute the following step (e) for L - times to obtain the scrambled image from the original image to be encrypted, where L=1 to 3, L=1 to obtain the scramble image and L=2 and 3 to re-scramble the scrambled image two times,

If $X\gamma_{SCB}(j) = 0$ → make theright circular shift

else → make the left circular shift … (12)

(6) If ITR = ITR_{max}, the image I_{ENC} is decrypted. Otherwise, the algorithm returns to step 2.

3 Results and Discussion

The conventional and modified algorithms were written using MATLAB tools. For the sake of both security and aesthetic evaluation, it is necessary to display the suggested adequately modified Rubik's cube encryption algorithm. We simulated both algorithms for 256x256 pixel grayscale pictures that were utilized for the visual examination. The simulation results of the conventional and modified Rubik's cube algorithm and the test images are shown in Fig. 3 and Fig. 4 respectively.

The modified Rubik's cube algorithm and its qualitative analysis are tabulated in Table 1 for

Table 1 — Qualitative analysis of existing Rubik's Cube[18] and Modified Rubik Cube algorithm.

Images	Methods	Quantitative Metrics											
		NPCR	UACI	MDMF	Entropy	SSIM	PSNR	Correlation Analysis - Input Image			Correlation Analysis - Decrypted Image		
								Horizontal	Vertical	Diagonal	Horizontal	Vertical	Diagonal
mandrill.jpg	Rubik's Cube	0.9960	6.821e-06	1.08E+05	7.6792	0.008	8.9827	0.9388	0.9469	0.8966	0.0279	0.0298	0.0161
	Modified Rubik's Cube	0.9963	0.3056	1.07E+05	7.6792	0.0068	8.9955	0.9434	0.9456	0.898	0.0124	0.0038	0.0039
lena.jpg	Rubik's Cube	0.9959	0.1289	8.69E+04	7.7599	0.011	8.647	0.9806	0.9609	0.9437	-0.0019	0.0005	-0.0042
	Modified Rubik's Cube	0.9960	0.31891	8.68E+04	7.7599	0.0084	8.6299	0.9803	0.9643	0.9428	0.0156	-0.0028	0.0149
onion.jpg	Rubik's Cube	0.9962	0.14467	3.96E+04	7.6197	0.0059	7.619	0.9861	0.9831	0.9702	0.0117	0.0001	-0.0038
	Modified Rubik's Cube	0.9962	0.2878	3.94E+04	7.6197	0.0061	7.6172	0.9834	0.9815	0.9732	-0.0023	-0.0185	0.0119
peppers.jpg	Rubik's Cube	0.9959	4.547e-06	6.88E+04	7.7448	0.0057	8.1022	0.9579	0.948	0.9157	-0.0145	0.0447	0.0014
	Modified Rubik's Cube	0.9961	0.31927	6.78E+04	7.7448	0.0074	8.1209	0.9588	0.9519	0.9205	-0.0061	0.0159	-0.0341
cameraman.tif	Rubik's Cube	0.9961	0.17281	6.43E+04	7.0097	0.0094	8.3959	0.9551	0.9416	0.894	-0.0193	-0.0236	0.0369
	Modified Rubik's Cube	0.9959	0.2965	6.46E+04	7.0097	0.0076	8.3576	0.9569	0.9338	0.9084	0.0212	0.0004	-0.0217
flower.jpg	Rubik's Cube	0.9960	6.747e-07	4.02E+05	7.1738	0.0094	8.7174	0.9461	0.9398	0.8892	-0.0094	-0.0184	-0.0072
	Modified Rubik's Cube	0.9962	0.2429	4.03E+05	7.1738	0.0092	8.703	0.9442	0.9331	0.9027	0.018	0.006	0.0084
lena.png	Rubik's Cube	0.9962	5.5051e-06	1.45E+05	7.3224	0.009	9.5037	0.9689	0.9377	0.9161	0.0043	-0.0245	0.0104
	Modified Rubik's Cube	0.9961	0.3144	1.46E+05	7.3224	0.0099	9.4789	0.9692	0.9457	0.9026	-0.0074	-0.0255	-0.0123
football.png	Rubik's Cube	0.9962	4.2126e-06	8.97E+04	6.6861	0.0079	8.2963	0.9541	0.9602	0.9503	0.0121	-0.0065	0.0108
	Modified Rubik's Cube	0.9964	0.2097	9.00E+04	6.6861	0.009	8.2657	0.9549	0.9678	0.9395	0.0121	-0.0017	-0.0115

comparison with the existing Rubik's cube algorithm[18]. Since, the Rubik's cube algorithm as mentioned in[18] is proposed only for grey-scaled images, and its corresponding output image for selected images such as coloured and grayscale images and its step-wise processed results are shown in Fig. 3 and modified Rubik's cube algorithm in Fig. 4. Meanwhile, Table 1 and Figs 3–4 show quantitative analyses of various image encryption and decryption metrics such as Unified Average Changing Intensity (UACI), Structural Similarity Index Measure (SSIM), Entropy, Number of Pixels Change Rate (NPCR), Maximum Deviation Measuring Factor (MDMF), and correlations. To address the shortcomings of the present Rubik's cube algorithm, we proposed a modified Rubik's cube algorithm, which is explained in section 3.2. It is clear from this graphic that the encrypted versions of the original photos seem completely different to the human eye.

3.1 Statistical Analysis

3.1.1 Number of Pixels Change Rate (NPCR)

The encrypted image should be significantly different from the original. To quantify this requirement, two different metrics are commonly utilized. The first metric is the number of pixels change rate (NPCR), which determines the proportion of distinct pixels between two images. Furthermore, NPCR's primary purpose is to assess whether a discriminatory assault occurred. Equation 13 calculates the fraction of different pixel counts between the planar and encrypted images.

$$\mathrm{NPCR} = \left(\frac{\sum_{i}^{M}\sum_{j}^{N} D(i,j)}{\mathrm{W} \times \mathrm{H}} \right) \times 100\% \quad \ldots (13)$$

Where W and H denote the encrypted images' width and height, respectively, and Images 1 and 2 represent the images before and after a pixel change, with two values, either 1 or 0, making them highly confusing.

3.1.2 Unified Average Changing Intensity (UACI)

The second approach, unified average changing intensity (UACI), computes the average intensity of pixel differences between two plain images and an encrypted image. The UACI is calculated by the Eq. 14

$$\mathrm{UACI} = \frac{1}{\mathrm{W} \times \mathrm{H}} \sum_{i}^{M}\sum_{j}^{N} \frac{\left| image1(i,j) - image2(i,j) \right|}{255} \times 100\% \quad \ldots (14)$$

To evaluate the performance of the best photo encryption algorithm, the NPCR should be as large as possible, with UACI values of around 33%. According to Table 1, the NPCR values in both the existing and proposed improved Rubik Cube

algorithms are greater than 99.5%, showing that the pixel variants were chosen at random.

3.1.3 Image Entropy (IE)

Entropy is a quantitative metric that provides uncertainty or complexity of a specific image. The entropy is represented in Eq.15:

$$H(x) = -\sum p(x)\, p(x) \quad \text{... (15)}$$

Where *p(x)* is the probability of source *x*. A greater image entropy metric score indicates more visual information. The lowest entropy value is zero, which occurs when the image pixel value remains constant in any position. The maximum amount of entropy for an image is determined by the number of grey scales. For example, for an image with 256 greyscale, the maximum entropy is log2(256) = 8. The maximum value occurs when all histogram bins have the same constant value, or when image intensity is equally distributed in [0, 255].

3.1.4 Correlation Coefficients Analysis (CCA)

Correlation coefficient analysis is a statistical process for determining the strength and direction of a linear relationship between two variables. It generates correlation coefficients ranging from -1 to 1. Positive correlations are denoted by positive values, negative correlations by negative numbers, and no associations by zeros.

3.1.4.1 Vertical and Diagonal

These terms are frequently used in a variety of contexts. "Diagonal" in the context of matrices usually refers to the elements that are located along the major diagonal of the matrix, which runs from top-left to bottom-right. Although "vertical" does not seem to be a typical word for matrices, it might be a column inside one. Because of their 8-bit nature, adjacent pixels in digital photography have a strong relationship. Pixel displacement is a valuable technique employed by picture encryption systems to address this issue. This study uses the correlation coefficient to assess the effectiveness of the encryption technique offered after the image has been deformed with the Arnold transform and the Lorenz chaotic sequence. A well-designed encryption method should produce an encrypted image with minimal correlation. This section compares the photographs before and after encryption, using Eq. 16.

$$r(X,Y) = \frac{\mathrm{cov}(X,Y)}{\sigma(X)\sigma(Y)} \quad \text{… (16)}$$

Pixel sequences X and Y are represented as follows: cov(X,Y) indicates X and Y's covariance, while σ(•) represents the standard deviation. r(X, Y), where Y is the pixel closest to each X in the horizontal, vertical, or diagonal directions, denotes the correlation coefficients for the horizontal, vertical, and diagonal axes. Next, r(X, Y) is used to calculate the correlation between two neighbouring pixels based on horizontal, vertical, and diagonal criteria. If r(X, Y) is close to one, it implies a significant correlation between adjacent pixels. However, when r(X, Y) approaches zero, there is little connection between adjacent pixels. To calculate the correlation coefficients for each orientation (horizontal, vertical, and diagonal), pairs of randomly chosen neighbouring locations were employed. Table 1 shows that there is a strong correlation between the pixels in the base image. However, there is no link between the adjacent pixels of the encrypted cypher image. The correlation coefficient of the original image is close to one, suggesting a strong connection in all directions, as shown in Table 1. However, the correlation coefficients for all orientations of the encrypted image are around zero, indicating that adjacent pixels are not closely connected. This demonstrates that our encryption strategy has a good encryption impact. Table 1 also reveals that our approach has a lower correlation coefficient than the reference technique for encrypting the image, meaning that the technology presented in this paper is more resistant to statistical attacks.

3.1.5 Peak signal-to-noise ratio (PSNR)

The image quality is estimated by objective quality assessments after any distortion, higher PSNR assesses the excellence of the image. The creation of quantifiable indicators that forecast perceived image quality is the main objective of objective evaluation. Mathematical models are objective evaluation tools. The most popular full reference (FR) objective quality metric for assessing the majority of image processing methods is the peak signal-to-noise ratio. The PSNR is defined as the ratio of a digital image's maximum element or pixel to its mean square error. Eqs.17–18 explain how to compute the MSE and PSNR respectively:

$$e^2ms = \frac{1}{N^2}\sum_i^N \sum_j^N (X(i,j) - Y(i,j))^2 \quad \text{… (17)}$$

$$PSNR = 10log_{10}\left(\frac{255^2}{e^2ms}\right) dB \quad \text{… (18)}$$

In grey scale photographs, the maximum pixel value is represented by MAX, which stands for 255. Methods of digital image processing, like encryption and decryption, are evaluated for performance using the PSNR. The initial image in digital photographs is the signal, whereas the encoded or decrypted image is measured noise. To prevent detectable discrepancies between the original and treated images, image processing algorithms typically use PSNR values ranging from 30 to 50 dB. A greater number is desired, for example, when compressing images because the produced image is closer to the original. Given that the two photos cannot be identical, the recommended encryption algorithms favour lower PSNRs between the original and encrypted images, whereas the suggested decoding approaches favour higher PSNRs.

3.1.6 The structural similarity index measure (SSIM)

An image quality metric that considers the visual impact of three image properties: brightness, contrast, and structure. The mean square error computes the average square error between each pixel in the two images under comparison. In contrast, SSIM looks for commonalities within pixels. i.e., whether the pixels in the two photos match and/or have similar pixel density values.An objective criterion of quality that contrasts the two photos' quality is the structural similarity measure (SSIM). A reference image that is original and unaltered and another that is twisted, encrypted, or compressed make up SSIM, which is a comprehensive reference evaluation. To optimize the outcomes of the suggested techniques, the degree of resemblance between the unencrypted and original photographs should be as great as it is practical, and vice versa. Calculate the brightness, contrast, and structure of both the X and Y images using Eqs.19–22, and then combine the findings to obtain the SSIM between the two images X and Y using Eq.22.

$$l(X,Y) = \frac{2\mu_x\mu_y+\mathrm{C1}}{\mu_x^2+\mu_y^2+C1} \quad \ldots (19)$$

$$c(X,Y) = \frac{2\sigma_x\sigma_y+\mathrm{C2}}{\sigma_x^2+\sigma_y^2+C1} \quad \ldots (20)$$

$$s(X,Y) = \frac{\sigma_{xy}+\mathrm{C3}}{\sigma_x\sigma_y+C3} \quad \ldots (21)$$

$$SSIM(X,Y) = l(X,Y)^{\alpha}.c(X,Y)^{\beta}.s(X,Y)^{\gamma} \quad \ldots (22)$$

The mean values of pictures X and Y are denoted by μ_x and μ_y, the variance values by σ_x and σ_y, and the covariance by σ_{xy}. C1, C2, and C3 are constants with small values that prevent the denominators when calculating l(X, Y), c(X, Y), and s(X, Y) from approaching zero. The constants C1, C2, and C3 are computed using the equations $C_1=(K_1L)^2$, $C_2=(K_2L)^2$, $C_3=C_2/2$, L The L value represents the dynamic range value of the pixels, which is 255 for 8-bit grayscale images, and K_1 and K_2 are 0.01 and 0.03, respectively. α, β, and γ must be greater than zero, and typically $\alpha = \beta = \gamma = 1$.

3.2 Security Analysis

Security in cryptography is a major concern. Strong picture encryption should be resistant to known plain text and cipher-text-only attacks, as well as brute-force and statistical analysis attacks. This section presents a security study of the recommended picture encryption technology. The security assessment is based on statistical analysis and key space analysis.

3.2.1 Key Generation

The key plays an important role in picture encryption and decryption. The most popular way to generate keys is to employ a random number or a specific technique, like the Pseudo Random Number Generator (PRNG), RSA (inventors: Rivest, Shamir, and Adelman) key generation, Linear Feedback Shift Register (LFSR), and so forth. The height and breadth of the image determine the key lengths for the Rubik's cube method. Assume that the P_C and P_R keys will be used in the procedure that is going to be described. For row operation, the width of the image and the length of the generated key must match. Likewise, for the column to function, the picture height and key length need to match. To put it another way, the image size and key length are the same. Values are picked from the set {0, 1, 2, 3, 4, ……., 2^{n-1}}, and α denotes the bit value of each key. The encryption process must function, and key selection during generation might be done deliberately or randomly.

A large key space is required for a safe image encryption method to make brute-force attacks practically (computationally) unfeasible. The proposed approach can theoretically accommodate an infinite key space. Nonetheless, the encryption key used in our technique consists of the triplet (P_R, P_C, ITR_{max}). The vectors PR and PC for an α-bit grayscale image Qo with X x Y pixels can theoretically have values of 2Xγ and 2Yγ, respectively. As the number of iterations of ITRmax increases, the key space can expand to 2γ(X+Y) x ITRmax x 22γ keys, assuming

that both vectors cannot have constant values. Consider the scale grey image of the original picture size versus the 8-bit picture with $ITR_{max} = 1$. Dimensions of the key space are 24097 - 216 ~ 101233. This key space can withstand prolonged attack and is bigger than the one used in photo encryption techniques. As previously mentioned, there has been a significant encryption change made to stages 4 (c to e) and 5 (c to e) of the suggested modified Rubik's cube method. The original image is used in these processes to create the scrambled image. Next, it is necessary to encrypt the image using the assumption that the row and column image pixel data for the scrambled image have L=1, and for the re-scrambled image, L=2 to 3. Considerable decryption modifications are also necessary for steps 4 (c to e) and 5 (c to e) since the original image is required to create the encrypted scrambled image. This means that for row and column image pixel data, L=1 for the scrambled image and L=2 to 3 for the re-scrambled image.

3.2.2 Key Sensibility

Furthermore, encryption systems should be particularly sensitive to changes in the encryption key, so that even minor changes to the key should result in a visible change in the encrypted or decrypted image. We carried out two studies to demonstrate the essential sensitivity of our scheme. The first one shows a substantial change in the photo encrypting approach. Here, the initial picture, Io, is encrypted with P1 = (P_R, P_C, ITRmax), using the randomly generated keys P_R, P_C, and ITRmax. The hundred iterations of this experiment are conducted utilizing distinct key pairs, P1 and P2, which only differ by the smallest amount. Subsequently, a second key, P2, is used to encrypt the same image or Qo. This key, P2 = (P_R, P_C, ITRmax+1), differs from the previous key, P1, in the least significant bit. Table 1 shows the mean and standard deviation of NPCR values derived from 100 different key pairings for the encrypted image with key P1 and the encrypted image with key P2. Tables (1-2) reveals that the average NPCR values are nearly 100%, indicating that the images encrypted with keys P1 and P2 differ significantly. The extremely low standard deviation statistics provide more evidence that the NPCR data are tightly packed around the mean. Figures 3 and 4 illustrate the original photographs, their encrypted versions using two distinct keys, P1 and P2, and the image difference between the encrypted images. As previously mentioned, there is just one bit that separates keys P1 and P2. The second test involves measuring the sensitivity of the key used to decrypt the image. To produce the encrypted image I_{ENC}, let's first encrypt the original image Io using the key P1 = (P_R, P_C, ITR max), where P_R, P_C, and ITR max are again selected at random. This image is decrypted individually using the keys P1 and P2, which never differ by more than one bit in the least significant bit area. Figures 3 and 4 show the original image, the encrypted image using key P1, the decrypted image using the correct key P1, and the decoded image with the erroneous key P2. This picture clearly shows that attempting to decode with the incorrect key will not work. Table 1 shows that the NPCR values for the Rubik's cube and modified Rubik's cube algorithms are nearly identical. All of the test photographs have values close to 0.99 or higher. This experimental investigation demonstrates that Rubick's and modified Rubick's algorithms have no negative impact on image quality. At the same time, comparing the SSIM and PSNR values in Tables 1–2 of the original and encrypted images reveals that there is no similarity because the SSIM numerical values are close to zero and the PSNR values are less than 10dB. When compared to the conventional algorithm the modified Rubick's cube algorithm is better in security at the same time it takes more iteration and requires a little more time to simulate the algorithm. The modified algorithm with colour

Table 2 — Qualitative analysis of Modified Rubik Cube algorithm for 256 x 256 colour images.

Images	Quantitative Metrics										
	NPCR	UACI	Entropy	SSIM	PSNR	Correlation Analysis - Input Image			Correlation Analysis - Decrypted Image		
						Horizontal	Vertical	Diagonal	Horizontal	Vertical	Diagonal
Yatch	0.9961	0.3212	7.6028	0.0099	8.4826	0.9265	0.9280	0.8612	-0.0020	-0.0032	5.955e-04
Cablecar	0.9959	0.3157	7.2714	0.0099	7.1236	0.9692	0.9601	0.9351	-0.0028	-1.574e-04	-8.911e-04
Cornfield	0.9960	0.3167	7.2911	0.0072	8.6158	0.8898	0.8592	0.8007	-0.0028	-0.0029	0.0034
Monarch	0.9959	0.3019	7.4723	0.0084	8.6773	0.8949	0.9305	0.8747	-0.0012	-9.876e-04	-0.0021
Zelda	0.9961	0.3120	7.7911	0.0076	8.5409	0.9680	0.9828	0.9549	-0.0044	0.0039	0.0011

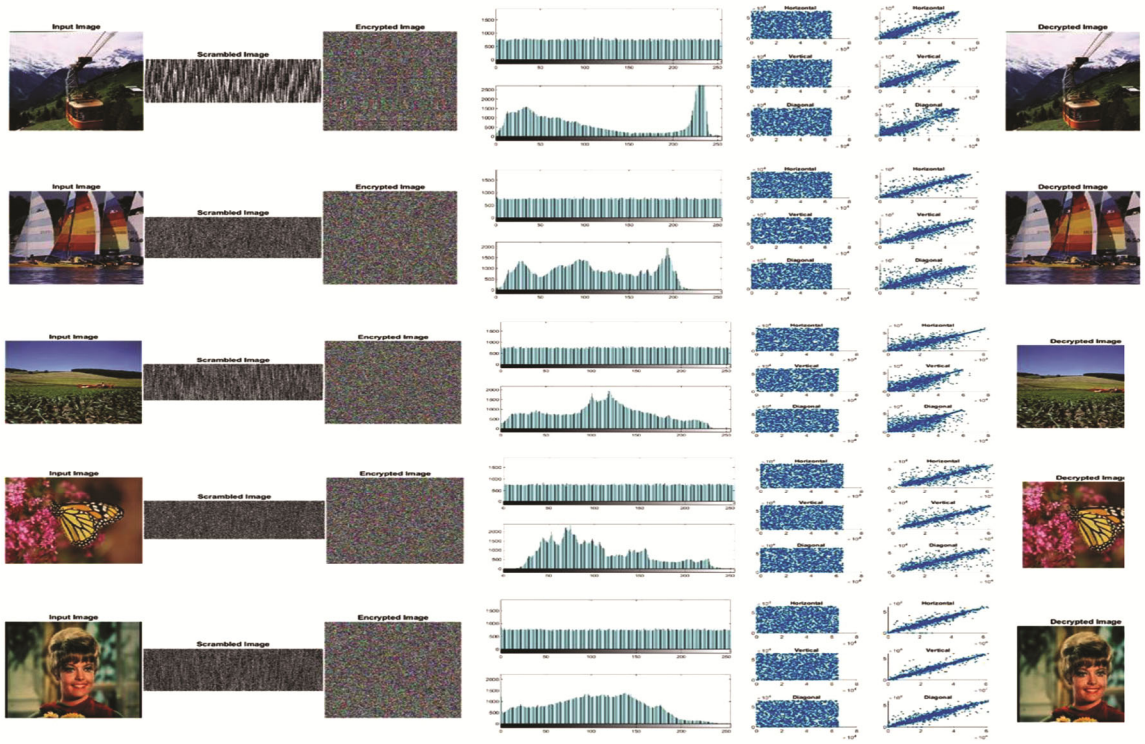

Fig. 5 — Modified Rubik's Cube Encryption and Decryption of colour images of size 256 × 256.

images and the performance matrices are presented in Fig. 5 and Table 2.

3.2.3 Execution Speed Calculation

From the implemented results in Tables 1 and 2, it is observed that the modified Rubick's cube algorithm achieves higher security for grey and colour images. Moreover, testing results show that the proposed technique does not sacrifice image quality and it is suitable for a wide range of multimedia applications. In real-time applications, the security is more concerned, at the same time how fast the given image is encrypted and decrypted by the modified algorithm. As we are aware most of the higher security algorithms are normally complex and require more time required to complete the execution. Many researchers are working on cryptography algorithm that requires less execution time and at the same time will provide higher security. The conventional and modified Rubik's cube were simulated in a 2.42GHz Intel processor with a RAM capacity of 8GB and the speed calculation was reported in Table 3. From Table 3, it is observed that both algorithm takes almost equal time when the smaller dimensional images. For the larger size image, the time required for the modified algorithm takes a marginally higher delay,

Table 3 — Execution speed calculation in seconds for the Lena image using a 2.4GHz processor with 8GB RAM personal computer.

Methods	Rubik`s cube		Modified Rubik`s cube	
Image Size	Encryption	Decryption	Encryption	Decryption
64 x 64	0.006059	0.005760	0.008382	0.006833
256 x 256	0.008309	0.008149	0.013027	0.011107
512 x 512	0.014832	0.017054	0.020633	0.017491

by implementing the algorithm in hardware [24-26] this can be highly reduced and will achieve almost equal time irrespective of the size of the image since the hardware-implemented algorithm requires time in terms of nano to pico seconds only compared to the seconds in software algorithms[27-28].

4 Conclusion

This work presents a photo encryption approach that may be used with both colour and grayscale images. This shareware uses the Rubik's cube idea to transform pixels in an image. When the XNOR operator is used to odd rows and columns of a picture using a key, the relationship between the original and encrypted images is obscured. To make the rows and columns even, flip the image and press the same key. A thorough numerical analysis was conducted, and the results show that the suggested approach is stable

against a wide range of attacks, including differential and statistical attacks (visual testing). Furthermore, testing results suggest that the modified Rubik's cube image encryption technique is accurate and secure for colour images when used to assess performance. Rubik's cube encryption technique and the modified Rubik's cube were successfully implemented on all recommended platforms due to their short processing time, demonstrating that even devices with limited computing capacity may use this technology. The speed of the algorithm increases predictably with image size, although the available mobile platforms are limited. Images with low resolution are handled quite skillfully. The mobile device's single core has enough processing power to encrypt many photos in a single second. This number rises since the majority of contemporary mobile devices feature multiple cores. According to the tests, the algorithm's speed is comparable to that of the advanced encryption standard technique. Subsequent studies will concentrate on enhancing the algorithm to hasten its implementation. This study presents an efficient and user-friendly recursive picture-scrambling technique based on the Rubik's Cube. When the procedure's initial settings which serve as the secret key are fed through a pseudo-random number generator, they yield a long sequence. Using the sensitivity of the first seed, the modified Rubik's cube securely generates random motions that confuse the image. It can be used in real-time photo encryption applications due to its near-linear linearithmic time complexity and recursive divide and conquer method. Because of the algorithm's flexibility, number sequences can be produced not just using the author's recommended number generator but also with logistic maps or other chaotic maps. The original image is always restored following the decryption procedure, and no information is lost in the process of encryption. Several experiments and security evaluations of the technique have demonstrated its robustness and suitability for jumbling every potential image in any real-world situation. In future to reduce the running time, the algorithm can be realised in hardware and implemented in field programmable gate array (FPGA) or intellectual property (IP) and integrated into the system on chip (SoC).

Funding: Not Applicable.

References

1 Lambić D, *J Inform Telecomm*, 2 (2018) 181.
2 Nair A, Dalal D & Mangrulkar R, *Cyber Sec Applic*, 2(2024) 1.
3 Kumari M, Gupta S & Sardana P, *3D Res*, 8 (2017) 1.
4 Agrawal M & Mishra P, *In J Comput Sci Eng*, 4 (2012) 877.
5 Govinda K & Prasanna S, *2015 International Conference on Soft-Computing and Networks Security (ICSNS), Coimbatore, India*, (2015) 1.
6 Ionescu V M & Diaconu A V, *2015 14th RoEduNet International Conference - Networking in Education and Research (RoEduNet NER), Craiova, Romania*, (2015) 121.
7 Raniprima S, Hidayat B & Andini N, *2016 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC), Bandung, Indonesia*, (2016) 198.
8 Joy J & Koshy L, *2019 IEEE International Conference on System, Computation, Automation and Networking (ICSCAN), Pondicherry, India,* (2019) 1.
9 Loukhaoukha K, Nabti M & Zebbiche K, *2013 8th International Workshop on Systems, Signal Processing and their Applications (WoSSPA), Algiers, Algeria*, (2013) 267.
10 Zhong H & Li G, *Multimed Tools Applic*, 81 (2022) 24757.
11 Patanwadia R & Mangrulkar R, *2021 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT), Bangalore, India*, (2021) 859.
12 Ge X, Lu B, Liu F & Gong D, *International Journal of Bifurcation and Chaos*, 26 (2016)
13 Zhang L, Tian X & Xia S, *2011 International Conference on Multimedia and Signal Processing, Guilin, China*, (2011) 312.
14 Helmy M, El-Rabaie E S M, Eldokany I M & El-Samie F E A, *3D Research*, 8 (2017) 1.
15 Gandhi S, Butani T & Gor R, *IOSR J Comput Eng (IOSR-JCE)*, 24 (2022) 68.
16 Mudduluri R V, Golla A, Raghava S & Sai T J, *Int J Eng Adv Technol (IJEAT)*, 11 (2022) 24.
17 Ionescu V M & Diaconu A -V, *2015 7th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), Bucharest, Romania,* (2015) P-31.
18 Loukhaoukha K, Chouinard J Y & Berdai A, *J Electr Comput Eng*, (2012) 1.
19 Abdullatif A A, Abdullatif F A & Naji S A, *Period Eng Natur Sci*, 7 (2019) 1607.
20 Vidhya R & Brindha M, *Journal of King Saud University - Computer and Information Sciences*, 34 (2022) 2000.
21 Xu L, Gou X, Li Z & Li J, *Opt Lasers Eng*, 91 (2017) 41.
22 Zhou Y, Bao L & Chen C L P, *Sign Proces*, 97 (2014) 172.
23 Zhang B & Liu L, *Mathematics*, 11 (2023)
24 Karuthapandian M, Banu M H, James B P & Dhandapani V, *Circuit World*, 47 (2021) 427.
25 James B P, Dhandapani V & Karuthapandian M, *Indian J Eng Mater Sci*, 27 (2020) 906.
26 Karuthapandian M, Marry R S &James B P, Dhandapani V, *Indian Journal of Pure & Applied Physics (IJPAP)*, 58 (2020) 605.
27 Britto J P, Vaithiyanathan D, *Int J Appl Eng Res*, 12 (2017) 2209.
28 Sharmila S, Bhuvaneswaran R S, Kailash N, Prathiba A, Vaithiyanathan D, *International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS 2023), M. P. Nachimuthu M. Jaganathan Engineering College, Erode, Tamil Nadu, India,* (2023) 1120.