An improved Chaos based Image Encryption Scheme Using 128-Bit Key

Deshdeepak Shrivastava[a], Dr Renu Bagoria[b] , Dr Aditya Vidyarthi[c]
[a]Phd Scholar,Faculty of Engineering ,Jagannath university ,Jaipur–302003India,
[b]Professor ,Faculty of Engineering, Jagannath university Jaipur-302003
[c]Professor ,Deprtment of Information Technology,ITM Gwalior-474001

Abstract: *With the rapid advancement of digital technologies, multimedia content such as images has become increasingly accessible and widely distributed. However, this growth has also raised significant concerns regarding the security and privacy of data during transmission and storage. In today's multimedia-driven world, images play a pivotal role in sectors like business, marketing, and digital promotions. Consequently, protecting image data from unauthorized access is of paramount importance. Image encryption serves as a crucial tool in the domain of information hiding. This article presents an enhanced chaos-based image encryption method that utilizes a 128-bit symmetric key for robust security. The effectiveness of the proposed encryption approach is evaluated using JDK1.7 and benchmarked against existing techniques. Experimental results demonstrate that the proposed scheme achieves higher entropy in the encrypted images, indicating improved security performance over traditional methods.*

Keywords: Encryption, Decryption, Chaos, Entropy, Pixel, Symmetric

1.INTRODUCTION

The proliferation of advanced digital technologies has significantly increased the accessibility and usage of multimedia data. As multimedia applications become increasingly integrated into everyday systems, ensuring the security of such data has become a key concern. Image encryption technology aims to convert raw visual content into an unintelligible format, ensuring that unauthorized individuals cannot interpret or misuse the data without the appropriate decryption key [1]. This approach plays a vital role in protecting images from unauthorized access, especially after they are downloaded by unidentified users. Government agencies use image encryption to securely manage confidential information and support covert operations [2]. In the healthcare field, encryption is essential to protect sensitive patient information. For example, medical records are often embedded with diagnostic images (e.g., X-rays) for secure, compact, and tamper-proof storage [3]. A key use of encrypted images is in remote mammography, where patient information and diagnostic data are securely embedded into the scan itself. This ensures both privacy and the integrity of the image during transmission and storage over long distances.

Image encryption can generally be divided into two main types. The first type embeds information into the image, making it hidden from unauthorized viewers. Accessing this hidden data requires decryption using a password. The second type, often referred to as public key encryption, focuses on encrypting the image itself rather than embedding external data in the image [4]. This method is mainly to prevent the image content from being tampered with, forged, and unauthorized changes. Among the common encryption strategies, the symmetric key encryption scheme uses a single key in the encryption and decryption process, thereby ensuring the confidentiality and integrity of the data.

The rest of this paper is arranged like this: Section 2 looks at related research. Section 3 explains the proposed design. Sections 4 and 5 describe how encryption and decryption work. Section 6 shares the results and what they mean. Section 7 gives the conclusion.

2.RELATED WORK

Amnesh Goel and others [5] suggested an image encryption method using the explosive block technique and showed that it works well for securely sending digital data over open networks.Their method focused on robust protection of visual content through systematic pixel operations. In another study, P. Karthigaikumar et al. [6] explored a chaos-based parameter modulation encryption method. The method only requires two iterations to enhance resistance to cryptographic attacks, thereby improving computational efficiency and alleviating the problem of dynamic performance degradation. Similarly, Manjunath Prasad et al. [7] proposed a chaos-based pixel scrambling algorithm. This technique uses the inherent randomness of the chaotic map to rearrange the pixel positions, thereby ensuring that the image data becomes unintelligible. During the decryption process, the original image is reconstructed by reversing the scrambling process in the order specified by the chaotic map.

Jolly Shah et al. [8] conducted a comprehensive classification of various image encryption techniques and evaluated them based on several key parameters, including scalability, visual quality degradation, compression compatibility, format compatibility, encryption rate, processing speed, and encryption robustness. [6] enhanced the previously proposed algorithm by randomly scattering the RGB components of each pixel throughout the image to enhance security. Zhang et al. [9] developed an encryption algorithm based on logical chaotic maps, which uses an 80-bit key combined with two chaotic logical maps to ensure encryption security. Another notable scheme was proposed in [10] , which uses a dynamic chaotic system to significantly improve the ability to resist cryptographic attacks. Specifically, a 16-byte key is used to choose one of three chaotic systems—Lorenz, Chen, or Lü— which is then applied to shuffle the positions of the image pixels [11]. At the same time, a second chaotic map from the same group is used to diffuse pixel values, thereby enhancing the complexity of the relationship between the plaintext image and the final ciphertext image.

The next section introduces an enhanced block-shift-based image encryption technique that rearranges the blocks horizontally and vertically as a preprocessing step before encryption. This bidirectional shift increases the randomness and complexity of the image structure, thereby improving security. The core encryption process then performs a shift-wise and XOR operation on the RGB components of each pixel using a 128-bit symmetric key. This combination of spatial and pixel-level transformations ensures stronger resistance to statistical and differential attacks.

3.PROPOSED ARCHITECTURE

Figure 1 illustrates the architecture of the proposed image encryption scheme. First, the input image is rescaled so that its dimensions in horizontal and vertical directions are exactly multiples of the predefined block length. This ensures uniform block processing. Next, inter-pixel shifting is performed by applying horizontal and vertical shifts to rearrange the pixel positions, thereby increasing the spatial complexity of the image. The altered image then moves to the encryption phase, where each 128-bit block of data is encrypted using a 128-bit symmetric key. This step continues until the entire image is fully encrypted. The final output is a fully encrypted image. To recover the original image, the decryption process is performed by reversing the encryption steps in the exact opposite order, thereby restoring the pixel arrangement and the original pixel values.

The HorizontalDisplacement and VerticalDisplacement methods are responsible for performing the directional rearrangement of the image pixels at the block level. In the HorizontalDisplacement method, blocks located at odd indices are swapped with the horizontal block in front of them, while blocks located at even indices are swapped with the horizontal block behind them. This symmetric swapping mechanism ensures that no data is lost during the displacement process as each swap is reversible. The number of pixels in each block is determined by the parameter BSize which defines the block size. The variables *rows* and *cols* indicate the number of rows and columns in the image, and they are used to correctly divide the image during the pixel displacement process.

The VerticalDisplacement method handles the vertical rearrangement of image blocks, and its functionality is similar to the HorizontalDisplacement method, with the main difference that the movement of blocks is performed along the vertical axis. During this process, blocks are systematically moved up and down according to their positions in the image matrix, thereby enhancing spatial diffusion. During the decryption phase, the inverse displacement operation is applied to restore the original image structure. Specifically, for vertical and horizontal displacements, odd-indexed blocks are swapped with the subsequent blocks, while even-indexed blocks are swapped with the previous block. This reversible block swapping mechanism ensures lossless recovery of the original image content during the decryption process.

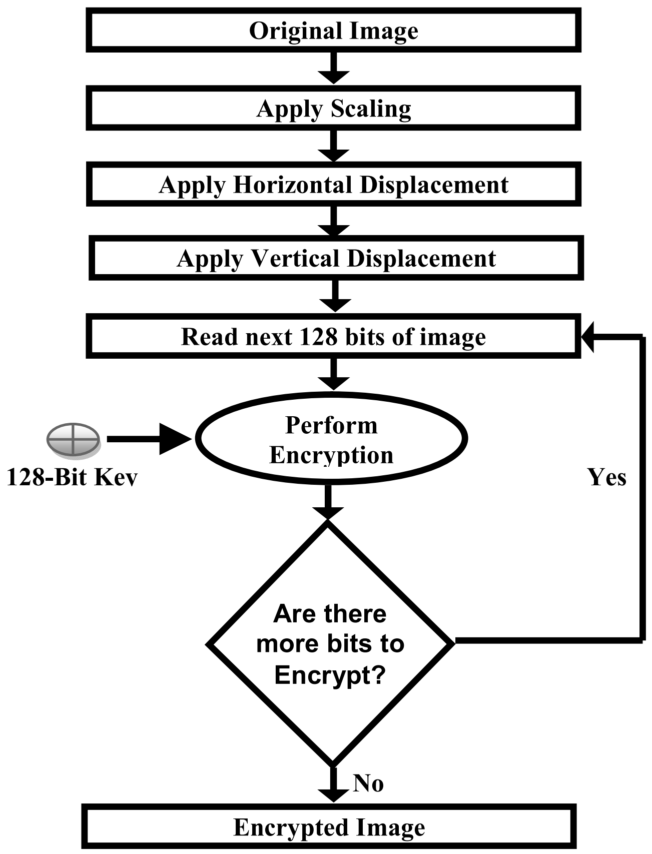

Original Image
Apply Scaling
Apply Horizontal Displacement
Apply Vertical Displacement
Read next 128 bits of image
Perform Encryption
128-Bit Kev
Yes
Are there more bits to Encrypt?
No
Encrypted Image

Fig.1. Architecture of proposed encryption scheme

ISSN NO: 1434-9728/2191-0073

4.PROPOSED ENCRYPTION PROCESS

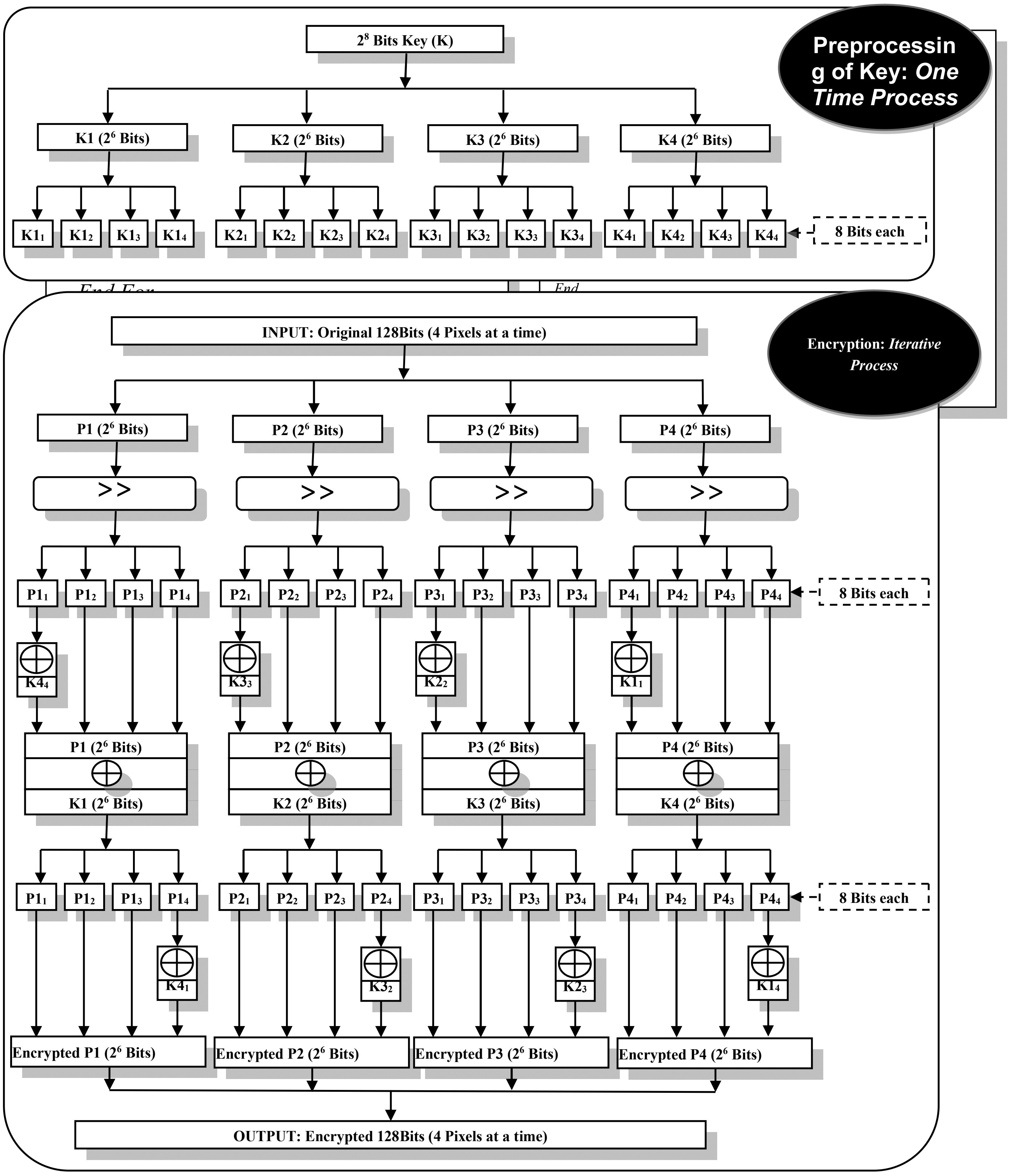


Fig.2. Proposed Encryption Process

At the initial stage of the encryption process, the original image is resized and divided into n × n uniform blocks for structured operations. Instead of encrypting the original image directly, the encryption process uses a scaled version, which helps optimize block-based operations. The block shifting technique plays a vital role in reducing the correlation between adjacent pixels and increasing the overall entropy of the image, thereby enhancing the encryption strength. To evaluate the effectiveness of the proposed method, we tested the algorithm with different block sizes. The function DisplaceAlgoHorizontal(image) calculates the total number of horizontal and vertical blocks in the image and applies a sequential displacement strategy along the horizontal axis. In this method, blocks are shifted one position to the right in a cascading manner: the block at position 1 is moved to position 2, block 2 is moved to position 3, and so on. This "explosive" displacement pattern significantly promotes pixel diffusion, further enhancing the randomness of the encrypted image.

Figure 2 presents the block diagram of the proposed encryption process, which is divided into two main modules: preprocessing of the encryption key and pixel encryption using the processed key. The process begins with preprocessing of the 128-bit key (K). Initially, the key *K* is split into four equal parts as *K1*, *K2*, *K3*, and *K4*, each 32 bits long. Then, each part *Ki* is further divided into four 8-bit segments: *Ki1*, *Ki2*, *Ki3*, and *Ki4*. This preprocessing step ensures the key is structured for efficient use during encryption. After the key preprocessing is completed, the image encryption process is as follows:
As illustrated in Figure 2, the encryption algorithm begins by preprocessing the 128-bit key (*K*) provided by the user. The key is divided into four equal segments: *K1*, *K2*, *K3*, and *K4*, each 32 bits in length. Each segment *Ki* is then further split into four 8-bit subparts: *Ki1*, *Ki2*, *Ki3*, and *Ki4*. After key preprocessing is complete, the image encryption process is as follows:
Read and preprocess the 128-bit key

- Divide the key into four 32-bit segments: *K1*, *K2*, *K3*, and *K4*
- Further split each segment *Ki* into four 8-bit parts: *Ki1*, *Ki2*, *Ki3*, and *Ki4*

Start reading the image in 128-bit blocks
Divide each 128-bit block into four equal parts

- Label the segments as *P1*, *P2*, *P3*, and *P4*, each 32 bits long

Right-shift each part (*P1*, *P2*, *P3*, and *P4*) by 2 bits
Further divide each 32-bit part *Pi* into four 8-bit segments

- Name them *Pi1*, *Pi2*, *Pi3*, and *Pi4*

Perform EX-OR operation between P11 and K44, and store the result in P11 as shown below:
$P1_1 = P1_1 \bigoplus K4_4$
Similarly update $P2_1$, $P3_1$ and $P4_1$ as follow:
$P2_1 = P2_1 \bigoplus K3_3$
$P3_1 = P3_1 \bigoplus K2_2$
$P4_1 = P4_1 \bigoplus K1_1$
Now perform EX-OR operation between 32 bit K1 and 32 bit P1, and store the result in P1 as follow:
$P1 = P1 \bigoplus K1$
Similarly update P2, P3 and P4 as follow:
$P2 = P2 \bigoplus K2$
$P3 = P3 \bigoplus K3$
$P4 = P4 \bigoplus K4$
Perform EX-OR operation between P11 and K44, and store the result in P11 as shown below:
$P1_4 = P1_4 \bigoplus K4_1$
Similarly update $P2_1$, $P3_1$ and $P4_1$ as follow:
$P2_4 = P2_4 \bigoplus K3_2$
$P3_4 = P3_4 \bigoplus K2_3$
$P4_4 = P4_4 \bigoplus K1_4$
Reform the 32 bits of P1 from $P1_1$,$P1_2$, $P1_3$ and $P1_4$
Reform the 32 bits of P2 from $P2_1$,$P2_2$, $P2_3$ and $P2_4$
Reform the 32 bits of P3 from $P3_1$,$P3_2$, $P3_3$ and $P3_4$
Reform the 32 bits of P4 from $P4_1$,$P2_2$, $P4_3$ and $P4_4$
The encrypted values labeled as P1, P2, P3, and P4 correspond to the transformed outputs of the initial four pixels. The process defined in Steps 2 through 8 should be iteratively applied until every pixel within the source image undergoes encryption.

5.PROPOSED DECRYPTION PROCESS

Fig.3 shows the block diagram of proposed decryption process. Once the pre-processing of key is over, decryption is done as follow:

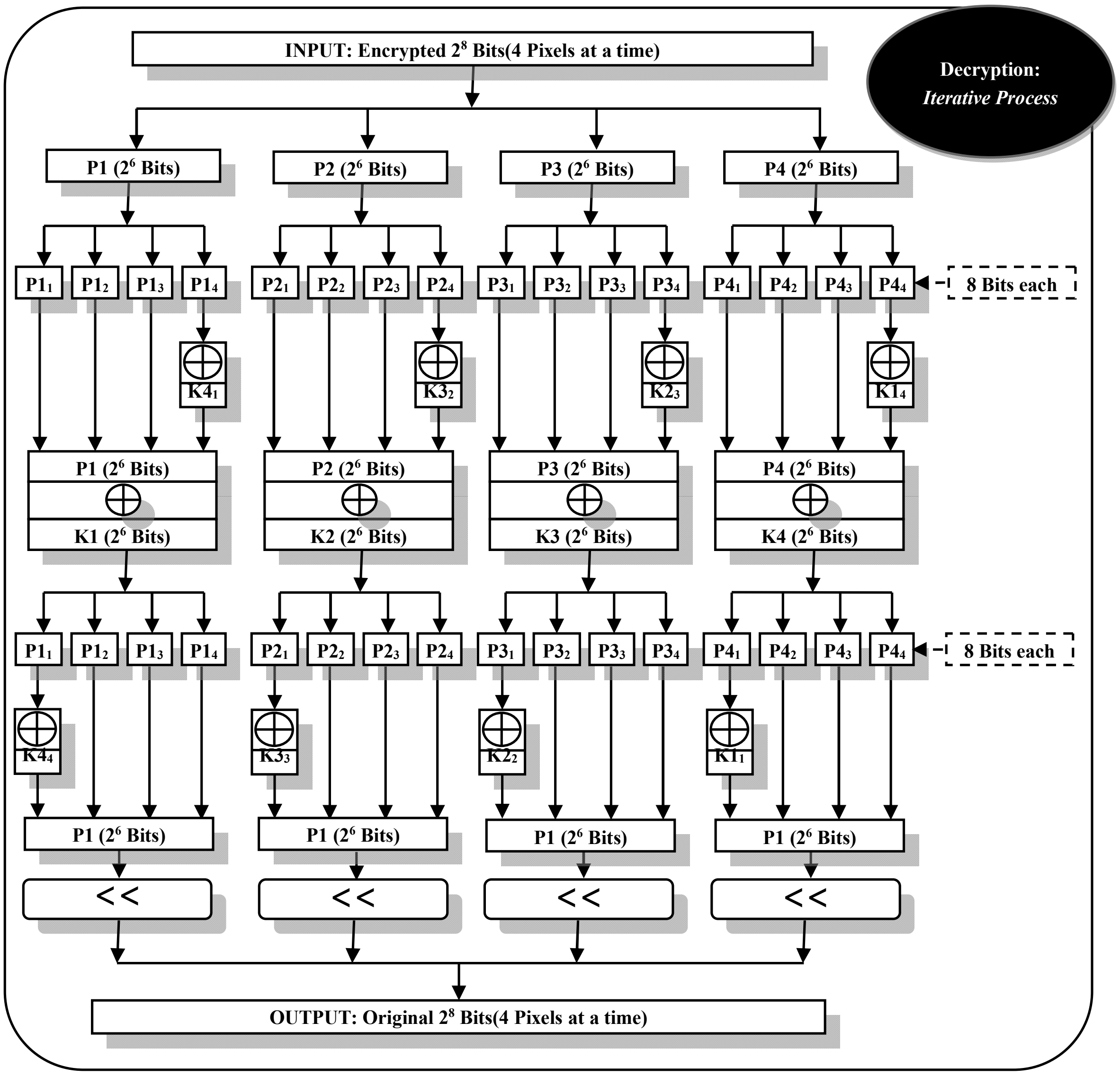


Fig.3. Proposed Decryption Process

Read and pre-process the 128-bit key
Start reading 128-bits of encrypted image at a time
Note: Since single pixel consists of 32 bits hence four pixels are read at a time as they contain 128 bits.
Divide the 128-bit data block into four equal segments: P1, P2, P3, and P4, where each segment holds 32 bits. Subsequently, partition each 32-bit segment (Pi) into four 8-bit units, denoted as Pi_1, Pi_2, Pi_3, and Pi_4, facilitating byte-level manipulation.
Perform *EX-OR* operation between $P1_1$ and $K4_4$, and store the result in $P1_1$ as shown below:

$P1_4 = P1_4 \oplus K4_1$
Similarly update $P2_1$, $P3_1$ and $P4_1$ as follow:
$P2_4 = P2_4 \oplus K3_2$
$P3_4 = P3_4 \oplus K2_3$
$P4_4 = P4_4 \oplus K1_4$
Proceed by executing a bitwise EX-OR operation between the 32-bit key segment K1 and the 32-bit data segment P1. The resulting 32-bit value should replace the original content of P1, effectively updating it with the encrypted output of this operation.:
$P1 = P1 \oplus K1$

Similarly update P2, P3 and P4 as follow:
$P2 = P2 \oplus K2$
$P3 = P3 \oplus K3$
$P4 = P4 \oplus K4$
Calculate *EX-OR* operation between $P1_1$ and $K4_4$, and save the result in $P1_1$ as shown below:
$P1_1 = P1_1 \oplus K4_4$
Similarly update $P2_1$, $P3_1$ and $P4_1$ as follow:
$P2_1 = P2 \oplus K3_3$
$P3_1 = P3_1\ K2_2$
$P4_1 = P4_1\quad K1_1$
Reform the 32 bits of P1 from $P1_1$,$P1_2$, $P1_3$ and $P1_4$
Reform the 32 bits of P2 from $P2_1$,$P2_2$, $P2_3$ and $P2_4$
Reform the 32 bits of P3 from $P3_1$,$P3_2$, $P3_3$ and $P3_4$
Reform the 32 bits of P4 from $P4_1$,$P2_2$, $P4_3$ and $P4_4$
The transformed values P1, P2, P3, and P4 represent the encrypted equivalents of the original four pixels after applying the encryption process. Apply 2 bits left shift on 32 bits of P1, P2, P3 and P4

Repeat Step.2 to Step.9 until all the pixels of input image are decrypted

6. EXPERIMENTAL RESULTS

Image entropy serves as a statistical indicator that quantifies the level of randomness or unpredictability within the texture of an image, reflecting the amount of information or complexity present in its pixel distribution.
The higher the entropy value, the stronger the randomness, which is desirable for encrypting images as it enhances security. To assess the effectiveness of the proposed encryption technique, the implementation was carried out using JDK 1.7 along with the Java Advanced Imaging (JAI) API. The entropy values of images encrypted through the proposed method were compared against those obtained using the inter-pixel and block shift encryption strategies introduced by Amnesh Goel et al. [5].

Fig.4. Entropy of images (640×400 P) encrypted with proposed scheme against existing scheme

$\oplus$
$\oplus$

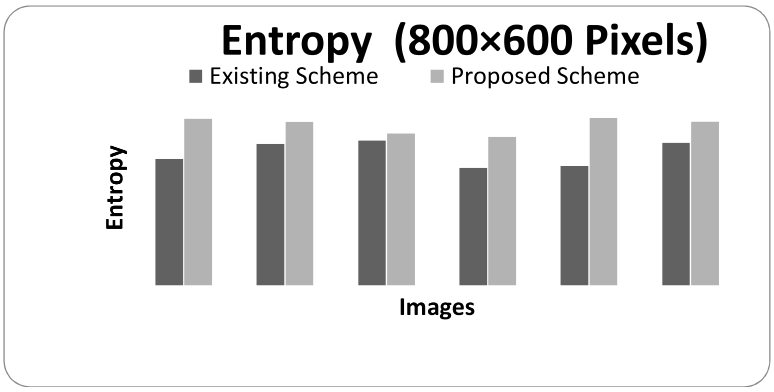
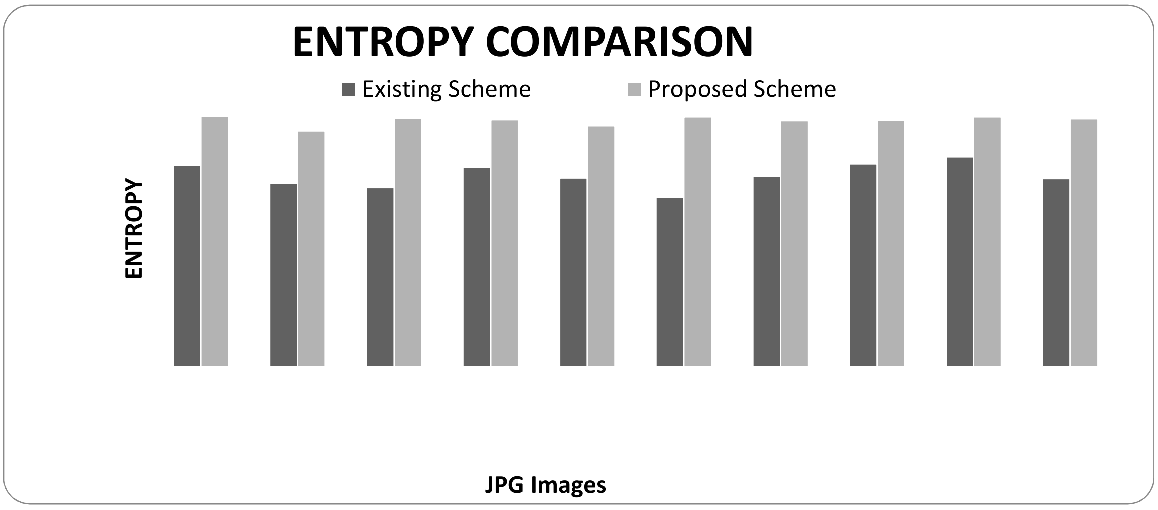


Fig.5. Entropy of images (800×600 P) encrypted with proposed scheme against existing scheme

Fig.6. Entropy of images (arbitrary size) encrypted with proposed scheme against existing scheme



The performance of the proposed encryption scheme was evaluated across three distinct image datasets. The first dataset consists of images with a resolution of 640×400 pixels, while the second includes images sized at 800×600 pixels. The third dataset features images of varying, non-uniform dimensions. These datasets encompass a diverse range of image characteristics, including differences in contrast and brightness levels. Additionally, the datasets contain both color and grayscale images, enabling a comprehensive assessment of the encryption scheme's robustness and adaptability.

Figures 4, 5, and 6 show the entropy comparison of images encrypted using the proposed scheme and images encrypted using the existing scheme. Figure 4 shows the entropy of a 640×400 pixel image, while Figures 5 and 6 show the entropy of 800×600 pixel and arbitrary size images, respectively. The graphical analysis reveals that the entropy values of images encrypted using the proposed scheme consistently surpasses those of images encrypted with existing methods. This indicates that the proposed encryption approach achieves greater randomness, thereby enhancing security irrespective of the image's size, contrast, brightness, or color composition.

Conclusion

In this paper, an enhanced image encryption scheme based on block shifting is proposed, which uses a 128-bit key. The performance of the proposed scheme is evaluated by comparing the entropy of the encrypted image with that of the image encrypted using the existing scheme. The results show that the proposed encryption method can introduce higher randomness, providing higher security than the existing methods, regardless of the size, contrast, brightness or color of the input image.

References:

[1] Choudhary, S., & Husain, S. (2023). Analysis of cryptography encryption for network security and image steganography technique. algorithms, 7(10).
[2] Van Daalen, O. L. (2023). The right to encryption: Privacy as preventing unlawful access. Computer Law & Security Review, 49, 105804.
[3] Islam, M. A. (2025). Enhancing Information Security Compliance in Healthcare through Cryptography and.
[4] Karthikeyan, B., Kishore, M. S., Sri Hari, R., Jaya Prakash, T., & Manikandan, G. (2024, July). Key Generation Technique by Password Hashing Using AES Encryption for Hiding Cipher Text within an Image. In 2024 Third International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT) (pp. 1-6). IEEE.
[5] Amnesh Goel and Nidhi Chandra "A Technique for Image Encryption Based On Explosive n*n Block Displacement Followed By Inter-Pixel Displacement of RGB Attribute of A Pixel" 2012 IEEE International Conference on Communication Systems and Network Technologies
6] P.Karthigaikumar, Soumiya Rasheed "Simulation of Image Encryption using AES Algorithm" IJCA Special Issue on "Computational Science - New Dimensions & Perspectives" NCCSE, 2011
[7] Manjunath Prasad1 and K.L.Sudha2and named "Chaos Image Encryption using Pixel shuffling " published in D.C. Wyld, et al. (Eds): CCSEA 2011, CS & IT 02, pp. 169–179, 2011.
[8] Jolly Shah and Dr. Vikas Saxena "Performance Study on Image Encryption Schemes" published in IJCSI International

Journal of Computer Science Issues, Vol. 8, Issue 4, No 1, July 2011 ISSN (Online): 1694-0814.
[9] Zhang, C., Chen, J., & Chen, D. (2022). Cryptanalysis of an image encryption algorithm based on a 2D hyperchaotic map. Entropy, 24(11), 1551.
[10] Francis, E. G., & Sheeja, S. (2023, December). Chaotic Resilience: Enhancing IoT Security Through Dynamic Data Encryption. In International Symposium on Intelligent Informatics (pp. 331-344). Singapore: Springer Nature Singapore.
[11] Lotfy, R. M. (2024). Hyperchaotic Chen System Based Visual Signal Encryption.