INDERSCIENCE PUBLISHERS
Linking academia, business and industry through research


**International Journal of Internet Technology and Secured Transactions**

ISSN online: 1748-5703 - ISSN print: 1748-569X
https://www.inderscience.com/ijitst

# Chaos-based block cipher for video stream encryption

Tran Quang Do, Nguyen Vu Ha Linh, Nguyen Hoang Minh, Ta Thi Kim Hue

**DOI:** 10.1504/IJITST.2025.10075131

**Article History:**

| | |
|---|---|
| Received: | 04 June 2025 |
| Last revised: | 24 September 2025 |
| Accepted: | 20 October 2025 |
| Published online: | 24 December 2025 |

# Chaos-based block cipher for video stream encryption

Tran Quang Do, Nguyen Vu Ha Linh, Nguyen Hoang Minh and Ta Thi Kim Hue*

School of Electrical and Electronic Engineering,
Hanoi University of Science and Technology (HUST),
No. 1 Dai Co Viet, Bach Mai,
Hanoi, Vietnam
Email: do.tq230010d@sis.hust.edu.vn
Email: linh.nvh213764@sis.hust.edu.vn
Email: minh.nh223724@sis.hust.edu.vn
Email: hue.tathikim@hust.edu.vn
*Corresponding author

**Abstract:** The growth of multimedia content in the digital era necessitates efficient and secure video encryption methods. Traditional algorithms like AES and DES often struggle with the computational demands of video data, especially in MP4 container formats. This paper proposes a novel chaos-based block cipher designed specifically for MP4 video encryption. Unlike prior methods relying on one-dimensional chaotic functions, this algorithm employs Arnold's cat map for robust pixel permutation and piecewise linear chaotic maps for diffusion, integrated within a multi-processing architecture. This approach ensures high security with low pixel correlation, high key sensitivity, and resistance to statistical and differential attacks. Our method balances security and efficiency, making it ideal for resource-constrained multimedia applications.

**Keywords:** chaos-based cryptography; video stream encryption; high dimensional chaotic maps, chaotic block cipher.

**Reference** to this paper should be made as follows: Do, T.Q., Linh, N.V.H., Minh, N.H. and Hue, T.T.K. (2025) 'Chaos-based block cipher for video stream encryption', *Int. J. Internet Technology and Secured Transactions*, Vol. 13, No. 7, pp.1–21.

**Biographical notes:** Tran Quang Do received his Master of Biomedical Engineering from Hanoi University of Science and Technology (HUST) in 2011. Currently, he has been a PhD student at HUST, Vietnam. He has over 20 years working in major hospitals in Vietnam and managing hospital information system and health equipment system. His current interests include multimedia, medical data security and privacy-preserving data in IoMT.

Nguyen Vu Ha Linh received her Bachelor's degree at HUST in 2025, Vietnam, majoring in Electronics and Telecommunications. She has been actively researching cryptography topics, including asymmetric encryption

and authentication techniques, for over two years at the Signal Information Processing Laboratory, HUST. Her research interests have recently expanded cryptography and multimedia security, with the goal of contributing to the advancement of information security.

Nguyen Hoang Minh is a fourth-year undergraduate student, majoring in Electronics and Telecommunications at HUST. For over two years, he has been actively involved in cryptography research at the Signal Information Processing Laboratory, HUST, Vietnam.

Ta Thi Kim Hue received her PhD in Electronic Engineering and Telecommunication Technologies from HUST, Vietnam in 2016. From 2015 to 2016, she was an exchange PhD student at the Department of Electronics and Informatics, Vrije Universiteit Brussel, Belgium. She has served as a lecturer at HUST since 2012. She is an (co-)author of more than than 30 international publications. She worked for almost 15 years as a researcher at the Signal Information Processing Laboratory, HUST, Vietnam. Her current interests include complex networks, cryptography, physical security protocols for the future internet, and preserving privacy data in e-healthcare.

## 1 Introduction

During the digital transformation era, the security of multimedia content has become increasingly important as the number of digital images and videos continues to grow exponentially. While effective for text data, traditional encryption algorithms such as AES and DES face challenges when applied to multimedia content, particularly in terms of processing efficiency and real-time performance requirements (Hosny et al., 2023). As a result of these challenges, intense research has been conducted to develop specialised encryption methods for multimedia data protection, with particular attention paid to maintaining synchronisation in video streaming applications and addressing the unique characteristics of multimedia data streams (Elkamchouchi et al., 2022).

Recent comprehensive surveys have highlighted the emergence of chaos-based cryptography as a viable method for multimedia encryption (Zia et al., 2022). These methods have demonstrated remarkable potential in achieving computational success and better security by integrating chaos theory with standardised encryption algorithms. Classifying chaos-based methods into spatial, temporal, and spatiotemporal domains has provided valuable insights into their applications and effectiveness. Furthermore, integrating chaos-based cryptosystems with public-key encryption algorithms like RSA has shown promising potential in enhancing overall system security while ensuring performance improvements (Hosny et al., 2023).

Contemporary research has extensively explored lightweight encryption methods using stream ciphers like ChaCha20 and hybrid chaotic maps, gaining enhanced encryption capability with reduced latency (Maolood et al., 2022). These approaches have been particularly successful in addressing the computational overhead associated with traditional encryption methods. However, many existing stream cipher approaches lack the robust security properties offered by block ciphers when processing the structured data found in MP4 video containers, which include both frame data and

metadata requiring comprehensive protection. The integration of Lévy flight with the rabbit lightweight stream cipher represents another considerable advancement, demonstrating robust resistance to common cryptanalysis attacks while providing real-time performance on smartphone platforms (Obaida et al., 2022). Similarly, the development of the SLEPX cipher for SHVC has established new benchmarks in both computational efficiency and format compliance while ensuring strong visual protection, particularly in scenarios requiring high-definition video content (Shah et al., 2020).

In the field of communications, chaos-based stream ciphers have been effectively utilised in VoIP systems to tackle critical issues of real-time speech encryption, significantly reducing packet loss and jitter while enhancing communication quality (Ibrahem and Qasim, 2021). Furthermore, advanced optical encryption methods have emerged as a promising direction, with the implementation of 3D-JST for bit-plane permutations in combination with 2D-FrFT encryption cascades showing increased resilience against both statistical and differential attacks (El-Shafai et al., 2021). The adoption of hyperchaotic Lorenz systems has propelled the field forward by allowing the selective encryption of crucial HEVC syntax elements, achieving an ideal balance between security needs and processing efficiency (Faragallah et al., 2022).

Current innovations have combined multiple approaches to achieve superior security and performance characteristics. The integration of logistic iterative chaotic maps with coupled map lattices (CML) for HEVC encryption has demonstrated encouraging outcomes in keeping video quality while maintaining robust security (Chen et al., 2021). Our work builds upon these foundations while addressing their limitations, particularly in regards to the specific challenges posed by MP4 container formats, which require different encryption strategies than raw video streams or HEVC-encoded content. Parallel developments in RC4-based approaches for network surveillance applications have demonstrated enhanced capabilities in handling video streams while maintaining security requirements (He, 2023). However, comprehensive cryptanalysis studies have revealed essential understandings into the security requirements of these systems, indicating that many existing approaches require a minimum of three rounds of encryption to reach adequate security against modern attack vectors (Lin et al., 2023).

These advancements collectively demonstrate the field's significant evolution toward more efficient and secure multimedia encryption solutions, though substantial obstacles persist in balancing work requirements with increasingly stringent security standards. The contributions from these diverse studies have not only enhanced our knowledge of multimedia encryption techniques but have also established new paradigms for their practical implementation in diverse scenarios. Looking forward, the integration of these various approaches and the development of hybrid solutions may offer promising directions for addressing the growing demands of secure multimedia communication in an increasingly connected world.

In this paper, we propose an effective block cipher algorithm designed specifically for MP4 video encryption. Our approach employs the Arnold's cat map for pixel permutation and chaotic functions for diffusion within a multi-round confusion-diffusion framework. We implement a multiprocessing architecture that enhances processing efficiency for various applications. Our experimental evaluation demonstrates the algorithm's robustness against statistical and cryptanalytic attacks, with strong performance in key security metrics such as high key sensitivity with number of pixels change rate (NPCR) and unified average changing intensity (UACI), low pixel correlation and resistance to differential and chosen-plaintext attacks. Compared to Jiang

et al. (2024), which utilised the Chirikov normal map for confusion, our adoption of the Arnold's cat map offers superior chaotic behaviour due to its two-dimensional mixing properties, which result in more uniform pixel permutation. This leads to enhanced security against statistical attacks, as the Arnold's cat map disrupts pixel patterns more effectively than one-dimensional sin-based transformations. Additionally, the Arnold's cat map's computational simplicity decreased processing overhead, making our method more efficient for real-time MP4 encryption, particularly when handling the complex structure of MP4 containers that include both video frames and metadata. These advantages make our approach better than existing methods in balancing security and performance, particularly for applications requiring robust protection of multimedia content in resource-constrained environments.

## 2 Principle of video encryption

Building upon the literature review presented above, it is evident that research in multimedia encryption has advanced significantly in recent years. This section delves deeper into the core principles of video encryption, building on the foundation of the techniques introduced previously and providing a more comprehensive analytical framework.

Video encryption extends beyond simply applying traditional encryption algorithms to video data; it requires consideration of the distinctive characteristics of modern video codec structures. When dealing with popular container formats like MP4, encryption approaches must account for the structured nature of these formats, which include distinct data sections for video streams and audio streams. Based on the relationship with compression, video encryption methods can be categorised into two main approaches (Liu and Koenig, 2010). Joint compression and encryption algorithms implement encryption during the compression process, which can be performed at various stages: after transformation (such as DCT or wavelet transform), after quantisation, or during entropy coding. The primary advantage of this approach is higher computational efficiency, especially appropriate for live applications and devices with limited resources. Conversely, compression-independent encryption algorithms apply encryption before or after the compression process, treating the video codec as a separate module. This approach offers flexibility and independence from specific codec formats but often incurs higher computational costs.

The effectiveness and security of video encryption depend significantly on the scope of data being encrypted. Complete encryption processes the entire video stream using conventional cryptographic algorithms such as AES or RSA. While this method provides the highest security, it faces substantial computational overhead, rendering it unsuitable for mobile or IoT devices. Selective or partial encryption targets only critical components such as I-frames, motion vectors, or specific DCT coefficients. These methods significantly reduce computational requirements while maintaining acceptable security levels. For MP4 files specifically, selective encryption can target key components such as sample tables (stbl), which contain critical information about media data locations, or the media data boxes (mdat) themselves, preserving the file structure while securing the content. The SLEPX method mentioned in the literature review represents an exemplary case of an effective selective encryption technique.

Content-aware encryption, a more advanced development from selective encryption, employs machine learning techniques to identify important components requiring protection. Region of interest (ROI)-based encryption utilises networks such as Faster R-CNN to identify and selectively encrypt important regions within video frames. Research by Duan et al. (2018) demonstrated that ROI typically constitutes only approximately 20% of each frame, allowing for an 80% reduction in encryption workload without compromising security (Liu and Koenig, 2010). Compared to lightweight techniques like ChaCha20 and rabbit discussed earlier, content-aware methods represent the latest trend in boosting security and performance.

The domain in which encryption operates has significant implications for both security and performance outcomes. Spatial domain techniques include pixel permutation and substitution methods. These approaches are straightforward to implement but often require higher computational costs for high-resolution videos. Frequency domain approaches target DCT coefficients through methods such as zigzag permutation or sign-bit modification. These techniques leverage the structure of modern compression codecs, typically being more efficient than spatial domain methods. Entropy coding encryption methods alter the compression process itself, using techniques such as multiple Huffman tables or randomised entropy coding. The primary advantage of these methods is minimal impact on compression efficiency. The combination of different domain techniques with stream cipher algorithms such as ChaCha20 and RC4, which were discussed previously, has proven effective in network surveillance and communication applications.

Many applications require encrypted video streams to remain decodable by standard decoders, leading to format-compliant encryption. This approach preserves the syntax structure of compressed video while encrypting information-carrying fields. This is achieved through techniques such as index-based encryption of variable-length codes, permutation and substitution of syntax units, and selective impact on HEVC/AVC syntax elements. For MP4 containers specifically, format-compliant encryption must preserve the hierarchical box structure that organises all data within the file, ensuring that file type boxes (ftyp), movie boxes (moov), and media data boxes (mdat) remain properly structured and accessible to standard players while the content itself is secured.

When evaluating the security of video encryption methods, several metrics are important: statistical analysis (evaluated through histogram equalisation and correlation reduction between adjacent pixels), differential attack resistance (measured NPCR and UACI values), key sensitivity tests (assessing output changes when keys are slightly modified), and perceptual security evaluation (analysing the extent to which visual information can still be extracted from encrypted video). Cryptanalysis has indicated that many encryption methods require a minimum of three rounds of encryption to achieve adequate security against modern attack vectors, as demonstrated in research on 6D chaos-based encryption (Lin et al., 2023). This requirement must be considered when implementing the lightweight encryption methods mentioned in the literature review.

The performance of video encryption solutions is typically evaluated based on several criteria: encryption overhead (computational cost relative to full encryption), real-time processing capabilities (impact on frame rate and latency), impact on compression efficiency (changes in file size and quality), and format compliance (compatibility with standard decoders). Different application scenarios demand different

balances of these factors. Entertainment applications may accept lower security for better performance, and mobile applications need to balance security, performance, and energy consumption.

As video applications continue to proliferate across diverse platforms and use cases, encryption principles must adapt accordingly. The optimal approach ultimately depends on specific requirements, with no single method suitable for all scenarios.

## 3 Chaos-based block cipher for video encryption

The proposed encryption scheme leverages a multi-process chaos-based approach. Unlike prior methods that rely on inverse transformations for confusion, our algorithm employs the Arnold's cat map for pixel permutation, combined with chaotic diffusion using byte sequences generated from piecewise linear chaotic maps (PLCM). The architecture is designed to ensuring both computational efficiency and robust security. The algorithm operates in several key stages:

1 Preprocessing: The input video frame is prepared for encryption by splitting it into three RGB colour planes to allow independent processing.

2 Chaotic system initialisation: The main process $P_m$ initialises the system using a pseudo random bit generator (PRBG) based on the PLCM, defined as:

$$u_{i+1} = F(u_i, \alpha) = \begin{cases} \frac{u_i}{\alpha}, & \text{if } 0 \le u_i < \alpha \\ \frac{u_i - \alpha}{0.5 - \alpha}, & \text{if } \alpha \le u_i \le 0.5 \\ F(1 - u_i, \alpha), & \text{if } 0.5 < u_i \le 1 \end{cases} \tag{1}$$

The key $K$ comprises two pairs $(\alpha_1, u_1)$ and $(\alpha_2, u_2)$, where $\alpha_1, \alpha_2 \in (0, 0.5)$ and $u_1, u_2 \in [0, 1]$, generated by a seeded random number generator to initialise two PLCMs. Each PLCM undergoes $r_\alpha$ pre-iterations to eliminate transient effects. The main PRBG produces parameters $\alpha_m^i$ for each process $P_a^i$, where $\alpha_m^i$ consists of two pairs $(\alpha, u_i)$ for two PLCMs, totaling 12 values for 3 processes (one for each colour plane). These are generated by iterating the first PLCM for even indices and the second PLCM for odd indices, with each process ensuring $\alpha \in (0, 0.5)$ by applying $\alpha = 1 - \alpha$ if $\alpha > 0.5$. A confusion seed $s_c$, a single integer, is derived by iterating the first PLCM once and converting the result to generate the Arnold's cat map parameters $m$ and $n$. Similarly, a diffusion seed array $s_d$, comprising $3 \times r$ byte values, is generated by iterating the first PLCM $3 \times r$ times, with each plane receiving $r$ values for diffusion initialisation. Each assistant process generates a byte sequence $B_i$ by iterating its two PLCMs and XORing the results for diffusion. From key $K$, the PRBG maps to $\alpha_m^i$ via alternating PLCM iterations, and to $s_c$ and $s_d$ through sequential iterations of the first PLCM, enabling randomised pixel shuffling and diffusion.

3 Confusion: Each colour plane (B, G, R) is processed by a dedicated process $P_a^i$ (with $i \in \{0, 1, 2\}$) using Arnold's cat map to permute pixel positions. The Arnold's cat map is defined as:

$$\begin{pmatrix} s' \\ t' \end{pmatrix} = Cat\_Funct \begin{pmatrix} s \\ t \end{pmatrix} \tag{2}$$

$$Cat\_Map \begin{pmatrix} s \\ t \end{pmatrix} = \begin{pmatrix} 1 & m \\ n & mn+1 \end{pmatrix} \begin{pmatrix} s \\ t \end{pmatrix} \bmod \begin{pmatrix} W \\ H \end{pmatrix} \tag{3}$$

where $(s, t)$ denotes the original pixel coordinate, $(s', t')$ is the permuted coordinate, and $m$ and $n$ are positive integers generated by the PRBG using $s_c$. $W$ and $H$ represent the width and height of the image frame, respectively. The modulo operations ensure that pixel coordinates remain within the finite image boundaries. The Arnold's cat map serves as a bijective transformation that exclusively performs pixel position permutation without modifying pixel values. In the encryption process, pixels are repositioned according to the transformation matrix with modulo operations constraining coordinates within image boundaries $[0, W-1] \times [0, H-1]$. The bijective property ensures that each pixel position maps to exactly one new position, maintaining the one-to-one correspondence essential for perfect restoration. During decryption, the inverse matrix operation with proper modular arithmetic precisely reverses the permutation, restoring each pixel to its original position regardless of image dimensions. This approach guarantees that the finite image space is properly handled while preserving the deterministic reversibility required for video decryption.

4 Diffusion: Each process $P_a^i$ generates a byte sequence $B_i$ from its PRBG and modifies pixel values in its component according to:

$$C_i^e[u, v] = k \oplus (C_i[u, v] + k) \oplus C_i^e[u', v'], \tag{4}$$

where $C_i^e$ is the encrypted colour plane, $C_i$ represents the colour plane being processed, $k$ is a byte from $B_i$, $[u, v]$ is the current pixel position, $[u', v']$ is the previous pixel position, and $\oplus$ denotes the XOR operation. The diffusion process starts with $C_i^e[u', v'] = s_d$ for the first pixel in each plane, creating a dependency across pixels that obscures the original content and strengthens security against statistical attacks.

5 Multi-round confusion and diffusion: To ensure robust encryption, these steps are repeated for $r$ iterators. In each round, every process performs the Arnold's cat map-based confusion followed by chaotic diffusion, with the main process $P_m$ collecting results by a queue to ensure all planes complete processing.

6 Recombination: The main process $P_m$ collects the encrypted planes (B, G, R) from all processes and merges them to reconstruct the full encrypted frame $F_e$, maintaining the original dimensions $W \times H$. The encrypted frame $F_e$ is then written to the encrypted video $V_e$, completing the encryption process for this frame.

The decryption process inverses the encryption steps to recover the original video, utilising the same multiprocess architecture but applying inverse operations for confusion and diffusion. It mirrors the encryption workflow in reverse order.

**Algorithm 1** Main process encryption

**procedure** EncryptMain($K, V$)
  **Input:** Key $K$, Video $V$
  **Output:** Encrypted video $V_e$
  Initialise $PRBG_m$ with key $K$
  **for** each frame $f$ in video $V$ **do**
    Split $f$ into colour planes: $C_0$ (B), $C_1$ (G), $C_2$ (R)
    Generate parameters from $PRBG_m$: $\alpha_m^0, \alpha_m^1, \alpha_m^2, s_c, s_d$
    Create processes $P_a^0, P_a^1, P_a^2$ for each colour plane
    **for** $r$ rounds **do**
      Perform confusion with input $C_i, s_c$ for all $i \in \{0, 1, 2\}$ in parallel
      Perform diffusion with input $C_i, s_d$ for all $i \in \{0, 1, 2\}$ in parallel
    **end for**
    Collect encrypted planes $C_0^e, C_1^e, C_2^e$ from processes
    $F_e = \text{Merge}(C_0^e, C_1^e, C_2^e)$
    Write $F_e$ to $V_e$
  **end for**
  **return** $V_e$
**end procedure**

**Algorithm 2** Subkeys generation

**procedure** GenerateSubkeys($\alpha_m^i, s_c, s_d, C_i$)
  **Input:** Parameters $\alpha_m^i, s_c, s_d$, colour plane $C_i$
  **Output:** Encrypted plane $C_i^e$
  Initialise $PRBG_a^i$ with $\alpha_m^i$
  **for** $r$ rounds **do**
    // **Confusion phase**
    Generate parameters of the Arnold's cat map $m, n$ from $PRBG_a^i$ using seed $s_c$
    $(s', t') = \text{Cat\_Map}(s, t, m, n)$ for each pixel at $(s, t)$ in $C_i$
    // **Diffusion phase**
    Generate byte sequence $B_i$ from $PRBG_a^i$
    Apply diffusion: $C_i^e[u, v] = k \oplus (C_i[u, v] + k) \oplus C_i^e[u', v']$,
      where $C_i^e[u', v'] = s_d$ for first pixel, $k \in B_i$
  **end for**
  $C_i^e = C_i$
  **return** $C_i^e$
**end procedure**

In the inverse diffusion phase, each assistant process $P_a^i$ (with $i \in \{0, \ldots, n-1\}$) applies the inverse diffusion operation to its colour plane $C_i^e$:

$$C_i[u, v] = (C_i^e[u, v] \oplus k \oplus C_i^e[u', v']) - k, \tag{5}$$

where $C_i^e[u, v]$ is the encrypted pixel in the colour plane, $C_i[u, v]$ is the recovered pixel, $k$ is a byte from the PRBG-generated sequence $B_i$, and $[u', v']$ is the previous pixel. The process traverses pixels in reverse, starting with the diffusion seed $s_d$.

The inverse confusion with the Arnold's cat map phase restores pixel positions using the inverse of the Arnold's cat map:

$$\begin{pmatrix} s \\ t \end{pmatrix} = InverseCat\_Map \begin{pmatrix} s' \\ t' \end{pmatrix} \tag{6}$$

$$InverseCat\_Map = \begin{pmatrix} mn+1 & -m \\ -n & 1 \end{pmatrix} \begin{pmatrix} s' \\ t' \end{pmatrix} \bmod \begin{pmatrix} W \\ H \end{pmatrix} \tag{7}$$

where $(s', t')$ is the permuted coordinates, $m$ and $n$ are encryption parameters. $W$ and $H$ are the image dimensions. The modulo operations wrap coordinates into the finite image space $[0, W-1] \times [0, H-1]$, make sure that the inverse transformation can properly map permuted pixels back to their original positions.

The multi-round inverse diffusion and confusion phase repeats inverse diffusion followed by inverse confusion for $r$ rounds. Finally, $P_m$ collects the decrypted planes (B, G, R) from all processes and merges them to reconstruct the full decrypted frame $F$. The frame $F$ is written to $V$, completing the decryption process.

**Algorithm 3** Main process decryption

**procedure** DECRYPTMAIN($K, V_e$)
  **Input:** Key $K$, encrypted video $V_e$
  **Output:** Decrypted video $V$
  Initialise $PRBG_m$ with key $K$
  **for** each encrypted frame $f_e$ in video $V_e$ **do**
    Split $f_e$ into encrypted colour planes: $C_0^e$ (B), $C_1^e$ (G), $C_2^e$ (R)
    Generate parameters from $PRBG_m$: $\alpha_m^0, \alpha_m^1, \alpha_m^2, s_c, s_d$
    Create processes $P_a^0, P_a^1, P_a^2$ for each colour plane
    **for** $r$ rounds **do**
      Perform inverse diffusion in parallel with input $C_i^e, s_d$ for all $i \in \{0, 1, 2\}$
      Perform inverse confusion in parallel with input $C_i^e, s_c$ for all $i \in \{0, 1, 2\}$
    **end for**
    Collect decrypted planes $C_0, C_1, C_2$ from processes
    $F = \text{MERGE}(C_0, C_1, C_2)$
    Write $F$ to $V$
  **end for**
  **return** $V$
**end procedure**

**Algorithm 4** Inverse subkeys generation

**procedure** INVERSEGENERATESUBKEYS($\alpha_m^i, s_c, s_d, C_i^e$)
  **Input:** Parameters $\alpha_m^i, s_c, s_d$, encrypted plane $C_i^e$
  **Output:** Decrypted plane $C_i$
  Initialise $PRBG_a^i$ with $\alpha_m^i$
  **for** $r$ rounds **do**
    // **Inverse diffusion phase**
    Generate byte sequence $B_i$ from $PRBG_a^i$
    Apply inverse diffusion:
      $C_i[u, v] = (C_i^e[u, v] \oplus k \oplus C_i^e[u', v']) - k$
      where $C_i^e[u', v'] = s_d$ for first pixel, $k \in B_i$
    // **Inverse confusion phase**
    Generate the Arnold's cat map parameters $m, n$ from $PRBG_a^i$ using seed $s_c$
    $(s, t) = \text{InverseCat\_Map}(s', t', m, n)$ for each pixel at $(s', t')$ in $C_i^e$
  **end for**
  $C_i = C_i^e$
  **return** $C_i$
**end procedure**

## 4 Experimental result and security evaluation

### *4.1 Histogram and chi-square*

A well-encrypted video should exhibit a nearly uniform histogram across its frames, effectively concealing the visual characteristics of the original content and thwarting statistical attacks. Parameters of the Arnold's cat map are chosen equal to $m = 11$, $n = 7$ (these values were illustrative test values used during algorithm development and demonstration) and the number of iterators $r = 5$. To evaluate this, we analysed the pixel distribution of encrypted frames from the video *Bus* for consistency. As shown in Figure 1, the histograms of encrypted frames demonstrate a uniform distribution, indicating a successful obfuscation of the original video content.

**Figure 1** Histograms of the original and encrypted frames (see online version for colours)

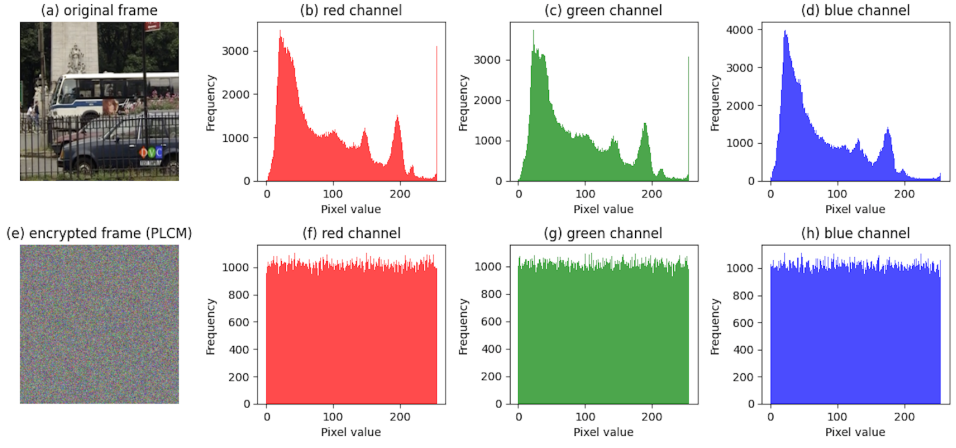
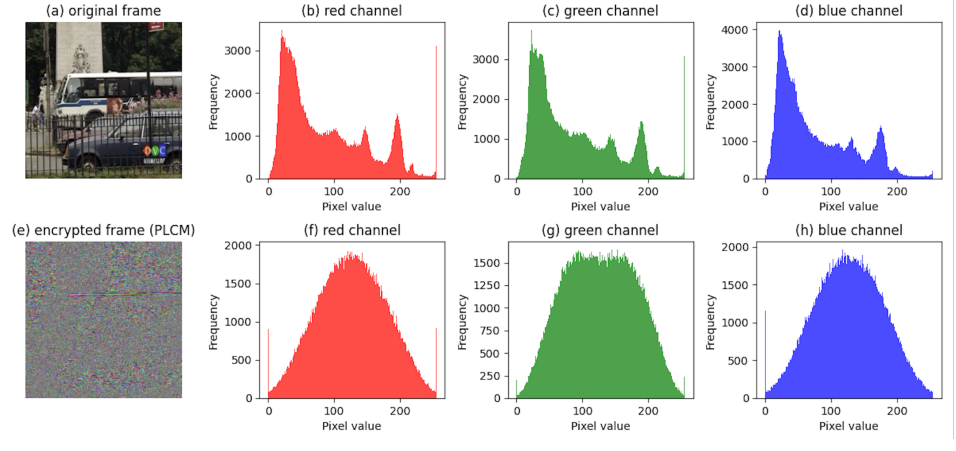



**Figure 2** Histogram of the original and encrypted frames in Jiang et al. (2024) (see online version for colours)



Compared with the work in Jiang et al. (2024) (the implementation using the source code available at https://github.com/jiangDongAHU/rtcve), our proposed method demonstrates superior histogram uniformity across all colour channels. While their approach shows several improvements over the original frame, quantitative analysis reveals significant limitations in achieving uniform distribution, as shown in Table 2. The encrypted video frame exhibits substantially higher variance values: red channel (363,441.102), blue channel (309,216.961), green channel (385,123.859). These variance values are approximately 300 times higher than our proposed algorithm, indicating inferior histogram uniformity. Moreover, all chi-square values exceed the theoretical threshold of 293.25. The histogram shows irregular distribution patterns with pronounced peaks and valleys, failing to achieve the uniform randomness required for robust cryptographic security. In contrast, the Arnold's cat map-based confusion mechanism produces more evenly distributed histograms with reduced variance in pixel intensity distributions.

To quantitatively assess histogram uniformity, we applied the chi-square test, defined as:

$$\chi^2_{\text{exp}} = \sum_{i=0}^{D-1} \frac{(o_i - a_i)^2}{a_i}, \tag{8}$$

where $o_i$ is the observed frequency of pixel intensity $i$ (0 to 255) in the histogram, and $a_i = \frac{W \times H \times N}{D}$ is the expected frequency, with $D = 256$.

For a truly random distribution, the chi-square value should be below the theoretical threshold of $\chi^2 = 293.25$ (with $\alpha = 0.05$). The results of our analysis on three different video sequences are summarised in Table 1.

**Table 1** Chi-square result

| *Video sequence* | *Red channel* | *Green channel* | *Blue channel* | *Average* | *Threshold* |
|---|---|---|---|---|---|
| Akiyo | 248.500 | 283.393 | 259.072 | 263.655 | 293.25 |
| Bus | 248.039 | 229.768 | 254.097 | 243.968 | 293.25 |
| Carphone | 250.697 | 230.293 | 277.811 | 252.934 | 293.25 |

**Table 2** Histogram quantitative comparison

| *Algorithm* | *Channel* | *Original variance* | *Encrypted variance* | *Original chi-square* | *Encrypted chi-square* | *Chi-square threshold* |
|---|---|---|---|---|---|---|
| Jiang et al. (2024) | Red | 720,681.734 | 363,441.102 | 180,170.4 | 90,860.3 | 293.25 |
| | Green | 769,099.969 | 309,216.961 | 192,275.0 | 77,304.2 | 293.25 |
| | Blue | 899,582.422 | 385,123.859 | 224,895.6 | 96,281.0 | 293.25 |
| Our work | Red | 720,681.734 | 992.156 | 180,170.4 | 248.0 | 293.25 |
| | Green | 769,099.969 | 919.070 | 192,275.0 | 229.8 | 293.25 |
| | Blue | 899,582.422 | 1,016.391 | 224,895.6 | 254.1 | 293.25 |

For all tested video sequences, the chi-square values remain below the theoretical threshold of 293.25. The average chi-square values across all three channels for each video also fall below this threshold. These results confirm that the encrypted videos exhibit uniform histograms, demonstrating the proposed algorithm's resistance to statistical attacks.

### 4.2 Differential attack resistance

Furthermore, we assessed the sensitivity of the encryption process by encrypting frames from MP4 video – *Bus*. Two encrypted versions of each frame were generated using keys differing by one bit, and their differences were evaluated using the NPCR and UACI, defined as follows:

$$\text{NPCR} = \frac{1}{W \times H \times N} \sum_{i=1}^{W} \sum_{j=1}^{H} \sum_{n=1}^{N} D(i,j,n) \times 100\% \tag{9}$$

where

$$D(i,j,n) = \begin{cases} 0, & \text{if } C_1(i,j,n) = C_2(i,j,n) \\ 1, & \text{otherwise} \end{cases} \tag{10}$$

$$\text{UACI} = \frac{1}{W \times H \times N \times 255} \sum_{i=1}^{W} \sum_{j=1}^{H} \sum_{n=1}^{N} |C_1(i,j,n) - C_2(i,j,n)| \times 100\% \tag{11}$$

Here, $W$, $H$, $N$ represent the width, height, and number of colour channels of a frame, respectively; $C_1(i,j,n)$ and $C_2(i,j,n)$ are pixel values at position $(i,j,n)$ in the two encrypted frames. To evaluate our method's performance relative to other approaches, Table 3 presents a comprehensive comparison of differential attack resistance metrics.

**Table 3** NPCR and UACI values comparison

| *Algorithm* | *NPCR (%)* | *UACI (%)* |
|---|---|---|
| Our work | 99.614 | 33.459 |
| Jiang et al. (2024) | 99.589 | 33.4635 |
| El-Shafai et al. (2021) | 99.635 | 33.565 |
| Maolood et al. (2022) | 99.716 | 33.725 |

Our results demonstrate competitive performance across all evaluated algorithms. These findings confirm that our block cipher achieves high key sensitivity and provides robust protection against differential attacks, validating its effectiveness as a secure video encryption method comparable to existing solutions.

### *4.3 NIST testing*

To ensure the suitability of the sequence generated by the cryptosystem for cryptographic applications, we employed the NIST SP 800-22 statistical test suite, a comprehensive framework comprising 15 distinct tests to detect non-randomness in binary sequences. Each test evaluates a specific aspect of randomness, producing a P-value that determines whether the sequence aligns with the hypothesis of ideal randomness at a significance level $\alpha$. A P-value $\geq \alpha$ indicates that the sequence is statistically random with high confidence ($\alpha = 0.01$).

In our experiments, we configured the algorithm to generate sequence using three assistant processes, denoted $P_a^i$ ($i \in \{1, 2, 3\}$), each producing byte sequences $B_i$ through the PLCM. These sequences were derived during the encryption of the video – *Bus*. Each test sequence contained 1,000,000 bits, with 10 independent sequences tested for each algorithm in ASCII format. The NIST test suite was configured with specific parameters in Rukhin et al. (2010): block frequency test used block length $M = 128$, non-overlapping template test employed block length $m = 9$, overlapping template test utilised block length $m = 9$, approximate entropy test applied block length $m = 10$, serial test used block length $m = 16$, and linear complexity test employed block length $M = 500$.

Table 4 summarises the results, showing that all sub-tests proportion consistently meeting the expected threshold for randomness. These outcomes confirm that the byte sequence produced by our system is sufficiently random, ensuring robust cryptographic security for video encryption applications.

**Table 4** Statistical test results

| *Statistic test* | *P-value* | *Proportion* |
|---|---|---|
| Frequency | 0.122325 | 9/10 |
| Block frequency | 0.534146 | 10/10 |
| Cumulative sums | 0.213309 | 10/10 |
| Runs | 0.350485 | 10/10 |
| Longest run | 0.739918 | 10/10 |
| Rank | 0.534146 | 10/10 |
| FFT | 0.122325 | 10/10 |
| Non-overlapping template | 0.911468 | 10/10 |
| Overlapping template | 0.066882 | 10/10 |
| Approximate entropy | 0.739918 | 10/10 |
| Serial | 0.911413 | 10/10 |
| Linear complexity | 0.350485 | 10/10 |

After conducting the NIST tests to evaluate the randomness of the output, we compared the results of our proposed method with several well-known stream ciphers, including RC4, Salsa20, Sosemanuk, and the results reported in Jiang et al. (2024). Table 5 provides a detailed comparison of the P-values obtained from the test. The results demonstrate that our method achieves competitive performance in multiple tests. These findings confirm the effectiveness and reliability of our proposed cryptosystem in generating high-quality random sequences suitable for cryptographic applications.

**Table 5** Statistical testing comparison (P-values)

| *Statistical test* | *Algorithm* | | | | |
|---|---|---|---|---|---|
| | *RC4* | *Salsa20* | *Sosemanuk* | *Jiang et al. (2024)* | *Our work* |
| Frequency | 0.534146 | 0.534146 | 0.122325 | 0.181557 | 0.122325 |
| Block frequency | 0.911413 | 0.911413 | 0.213309 | 0.474986 | 0.534146 |
| Cumulative sums | 0.350485 | 0.534146 | 0.911413 | 0.452541 | 0.213309 |
| Runs | 0.350485 | 0.534146 | 0.739918 | 0.350485 | 0.350485 |
| Longest run | 0.122325 | 0.739918 | 0.739918 | 0.883171 | 0.739918 |
| Rank | 0.534146 | 0.911413 | 0.534146 | 0.366918 | 0.534146 |
| FFT | 0.122325 | 0.534146 | 0.911413 | 0.897763 | 0.122325 |
| Overlapping template | 0.534146 | 0.739918 | 0.350485 | 0.366918 | 0.066882 |
| Approximate entropy | 0.213309 | 0.122325 | 0.911413 | 0.983453 | 0.739918 |
| Serial | 0.739918 | 0.350485 | 0.534146 | 0.428515 | 0.911413 |
| Linear complexity | 0.534146 | 0.350485 | 0.534146 | 0.137282 | 0.350485 |

### *4.4 Correlation analysis*

To assess the algorithm's ability to disrupt pixel dependencies, we analysed correlations between adjacent pixels in encrypted video frames. Frames from *Bus* were used. For 100 frames per video, 10,000 adjacent pixel pairs (horizontal, vertical, diagonal) were sampled per channel, and correlation coefficients were computed:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}. \tag{12}$$

Our result shows original frames with high correlations and encrypted frames with near-zero correlations, confirming the effectiveness in eliminating pixel relationships for secure video encryption.

**Table 6** Average correlation coefficients results

| *Direction* | *Original frame* | *Encrypted frame* |
|---|---|---|
| Horizontal | 0.968119 | 0.002351 |
| Vertical | 0.980419 | 0.006372 |
| Diagonal | 0.944089 | 0.002434 |

### *4.5 Parameter sensitivity*

To validate the robustness of our Arnold's cat map implementation regarding parameter selection, we conducted comprehensive experiments across multiple parameter combinations and image resolutions. This analysis indicates that the security of the encryption scheme derives from the intrinsic mathematical properties of the chaotic map rather than from specific parameter values. Although chaotic maps are highly sensitive to parameter variations, these changes do not compromise the cryptographic strength of the scheme, thereby ensuring a high level of reliability.

We systematically evaluated 30 different $(m, n)$ parameter pairs: (1, 2), (2, 3), (3, 5), (4, 5), (4, 6), (5, 7), (6, 8), (7, 11), (8, 10), (10, 12), (11, 13), (12, 13), (13, 15), (14, 16), (15, 17), (16, 17), (17, 19), (18, 20), (19, 21), (20, 22), (21, 23), (23, 25), (24, 26), (25, 27), (28, 30), (29, 31), (30, 32), (32, 36), (33, 35), (38, 45) across three image resolutions using the standard Akiyo video frame image. Each configuration was tested with $r = 5$ rounds of confusion-diffusion operations. The results of chi-square, average correlation coefficients, NPCR (%), PSNR (dB), wPSNR (dB), MSE and SSIM are presented in Tables 7, 8 and 9.

**Table 7** Encryption metrics for Akiyo 256 × 256

| *(m, n)* | *Chi-square* | *Avg. correlation coefficients* | *NPCR (%)* | *PSNR (dB)* | *wPSNR (dB)* | *MSE* | *SSIM* |
|---|---|---|---|---|---|---|---|
| (1, 2) | 259.3932 | 0.000597 | 99.6007 | 8.0238 | 8.2000 | 10,249.5313 | 0.0192 |
| (2, 3) | 245.1510 | 0.000374 | 99.6155 | 7.9855 | 8.1629 | 10,340.2591 | 0.0170 |
| (3, 5) | 250.7526 | 0.001371 | 99.6063 | 8.0080 | 8.1718 | 10,329.8149 | 0.0182 |
| (4, 5) | 230.4974 | 0.001009 | 99.5982 | 7.9901 | 8.1692 | 10,329.3759 | 0.0193 |
| (4, 6) | 263.0625 | 0.002763 | 99.6170 | 8.0017 | 8.1713 | 10,301.7959 | 0.0191 |
| (5, 7) | 237.7474 | 0.001994 | 99.5956 | 8.0007 | 8.1806 | 10,303.9999 | 0.0190 |
| (6, 8) | 252.362 | 0.000467 | 99.6277 | 8.0125 | 8.1881 | 10,276.0319 | 0.0183 |
| (7, 11) | 245.8047 | 0.000039 | 99.6338 | 8.0183 | 8.1856 | 10,262.4124 | 0.0174 |
| (8, 10) | 227.0078 | 0.000801 | 99.6185 | 8.0228 | 8.1972 | 10,251.8945 | 0.0205 |
| (10, 12) | 239.9010 | 0.001700 | 99.6078 | 7.9903 | 8.1638 | 10,328.8576 | 0.0196 |
| (11, 13) | 257.8438 | 0.000734 | 99.5936 | 7.9710 | 8.1467 | 10,374.8158 | 0.0167 |
| (12, 13) | 257.1120 | 0.000400 | 99.6099 | 8.0014 | 8.1708 | 10,302.5178 | 0.0177 |
| (13, 15) | 258.8958 | 0.001435 | 99.6048 | 7.9870 | 8.1610 | 10,336.6814 | 0.0164 |
| (14, 16) | 228.1745 | 0.000955 | 99.6028 | 7.9870 | 8.1619 | 10,336.6129 | 0.0177 |
| (15, 17) | 240.3594 | 0.002244 | 99.6150 | 8.0019 | 8.1773 | 10,301.3769 | 0.0173 |
| (16, 17) | 226.4557 | 0.002852 | 99.5987 | 7.9806 | 8.1567 | 10,351.8865 | 0.0192 |
| (17, 19) | 236.2604 | 0.001343 | 99.5880 | 8.0026 | 8.1826 | 10,299.5202 | 0.0184 |
| (18, 20) | 265.1276 | 0.000881 | 99.5941 | 7.9905 | 8.1707 | 10,328.4024 | 0.0191 |
| (19, 21) | 259.2682 | 0.000795 | 99.6333 | 7.9901 | 8.1690 | 10,329.2024 | 0.0178 |
| (20, 22) | 257.2682 | 0.000997 | 99.6104 | 8.0247 | 8.1894 | 10,247.4308 | 0.0184 |
| (21, 23) | 254.2708 | 0.002052 | 99.6109 | 8.0038 | 8.1683 | 10,296.7791 | 0.0172 |
| (23, 25) | 259.8646 | 0.001520 | 99.6185 | 8.0117 | 8.1802 | 10,277.9933 | 0.0190 |
| (24, 26) | 270.2552 | 0.000568 | 99.5911 | 8.0004 | 8.1699 | 10,304.7796 | 0.0194 |
| (25, 27) | 246.7370 | 0.000322 | 99.6084 | 8.0049 | 8.1756 | 10,294.1247 | 0.0190 |
| (28, 30) | 250.0911 | 0.000048 | 99.6333 | 7.9834 | 8.1644 | 10,345.1410 | 0.0157 |
| (29, 31) | 254.2057 | 0.000811 | 99.5951 | 8.0183 | 8.1957 | 10,262.4248 | 0.0181 |
| (30, 32) | 251.6641 | 0.000281 | 99.6048 | 7.9868 | 8.1690 | 10,337.1243 | 0.0176 |
| (32, 36) | 237.2943 | 0.001445 | 99.6429 | 8.0085 | 8.1888 | 10,285.7065 | 0.0178 |
| (33, 35) | 272.8568 | 0.002133 | 99.5860 | 8.0018 | 8.1822 | 10,301.5273 | 0.0194 |
| (38, 45) | 261.9375 | 0.001154 | 99.5855 | 7.9939 | 8.1714 | 10,320.3254 | 0.0176 |

**Table 8** Encryption metrics for 512 × 512

| *(m, n)* | *Chi-square* | *Avg. correlation coefficients* | *NPCR (%)* | *PSNR (dB)* | *wPSNR (dB)* | *MSE* | *SSIM* |
|---|---|---|---|---|---|---|---|
| (1, 2) | 252.0664 | 0.000684 | 99.6007 | 7.9837 | 8.1596 | 10,344.5814 | 0.0190 |
| (2, 3) | 266.3906 | 0.000793 | 99.6142 | 7.9882 | 8.1609 | 10,333.8419 | 0.0195 |
| (3, 5) | 262.1165 | 0.000555 | 99.5959 | 7.9954 | 8.1695 | 10,316.7820 | 0.0185 |
| (4, 5) | 266.7025 | 0.000715 | 99.6140 | 7.9766 | 8.1489 | 10,361.4527 | 0.0184 |
| (4, 6) | 253.9655 | 0.000910 | 99.6034 | 7.9789 | 8.1554 | 10,356.0124 | 0.0180 |
| (5, 7) | 264.6615 | 0.000726 | 99.6100 | 7.9776 | 8.1513 | 10,358.9482 | 0.0186 |
| (6, 8) | 265.5410 | 0.000327 | 99.6127 | 7.9881 | 8.1650 | 10,333.9394 | 0.0188 |
| (7, 11) | 255.0384 | 0.000345 | 99.6015 | 7.9808 | 8.1557 | 10,351.4679 | 0.0181 |
| (8, 10) | 258.9336 | 0.000721 | 99.6175 | 7.9876 | 8.1575 | 10,335.2019 | 0.0193 |
| (10, 12) | 235.4121 | 0.000582 | 99.6234 | 7.9980 | 8.1707 | 10,310.5579 | 0.0191 |
| (11, 13) | 251.5052 | 0.001026 | 99.6211 | 7.9903 | 8.1697 | 10,328.8233 | 0.0193 |
| (12, 13) | 279.3724 | 0.000962 | 99.6122 | 7.9871 | 8.1610 | 10,336.4879 | 0.0186 |
| (13, 15) | 235.7799 | 0.000019 | 99.6206 | 7.9776 | 8.1488 | 10,358.9869 | 0.0193 |
| (14, 16) | 262.5755 | 0.000146 | 99.6070 | 7.9756 | 8.1501 | 10,363.7371 | 0.0183 |
| (15, 17) | 255.7702 | 0.001298 | 99.6010 | 7.9824 | 8.1624 | 10,347.7118 | 0.0186 |
| (16, 17) | 253.8516 | 0.000547 | 99.6222 | 7.9903 | 8.1678 | 10,328.9146 | 0.0182 |
| (17, 19) | 288.7188 | 0.000842 | 99.6195 | 7.9932 | 8.1721 | 10,321.8272 | 0.0192 |
| (18, 20) | 244.8522 | 0.000661 | 99.6031 | 7.9902 | 8.1652 | 10,329.1442 | 0.0188 |
| (19, 21) | 248.2747 | 0.000979 | 99.5975 | 7.9854 | 8.1601 | 10,340.4962 | 0.0193 |
| (20, 22) | 244.7533 | 0.000126 | 99.6025 | 7.9930 | 8.1685 | 10,322.4169 | 0.0190 |
| (21, 23) | 246.9447 | 0.000244 | 99.6086 | 7.9879 | 8.1615 | 10,334.4786 | 0.0189 |
| (23, 25) | 242.8053 | 0.000741 | 99.6099 | 7.9894 | 8.1616 | 10,330.8557 | 0.0191 |
| (24, 26) | 237.1836 | 0.001284 | 99.6230 | 7.9890 | 8.1663 | 10,331.9908 | 0.0194 |
| (25, 27) | 258.2383 | 0.000589 | 99.6179 | 7.9864 | 8.1607 | 10,338.1916 | 0.0187 |
| (28, 30) | 268.3965 | 0.000567 | 99.6061 | 7.9918 | 8.1664 | 10,325.2263 | 0.0187 |
| (29, 31) | 246.4753 | 0.000430 | 99.5981 | 7.9899 | 8.1657 | 10,329.6484 | 0.0182 |
| (30, 32) | 233.8984 | 0.000048 | 99.6042 | 7.9737 | 8.1587 | 10,368.3427 | 0.0191 |
| (32, 36) | 253.3223 | 0.000907 | 99.6068 | 7.9761 | 8.1563 | 10,362.7007 | 0.0190 |
| (33, 35) | 250.5182 | 0.000636 | 99.6118 | 7.9906 | 8.1631 | 10,328.1518 | 0.0192 |
| (38, 45) | 261.1634 | 0.000149 | 99.5984 | 7.9877 | 8.1667 | 10,335.0336 | 0.0183 |

To ensure statistical reliability, 95% confidence intervals were calculated for each performance metric across all tested image resolutions in Table 10. These confidence intervals provide a range of values within which the true population parameter is likely to fall with 95% certainty. The intervals were computed using the standard error of the mean and appropriate critical values, allowing for robust statistical interpretation of the encryption algorithm's performance.

The narrow confidence intervals across all security metrics demonstrate that encryption quality is statistically independent of specific $(m, n)$ values. All chi-square values remain well below the threshold of 293.25, confirming uniform histogram distribution regardless of parameter selection. Moreover, all quality metrics (PSNR, wPSNR, MSE, SSIM) exhibit minimal variance across parameter combinations, confirming that encryption effectiveness is maintained regardless of the specific $(m, n)$ values used.

**Table 9** Encryption metrics for Akiyo 1,024 × 1,024

| *(m, n)* | *Chi-square* | *Avg. correlation coefficients* | *NPCR (%)* | *PSNR (dB)* | *wPSNR (dB)* | *MSE* | *SSIM* |
|---|---|---|---|---|---|---|---|
| (1, 2) | 255.3942 | 0.000467 | 99.6099 | 7.9967 | 8.1741 | 10,313.5986 | 0.0192 |
| (2, 3) | 249.4712 | 0.000459 | 99.6097 | 7.9838 | 8.1579 | 10,344.3248 | 0.0191 |
| (3, 5) | 250.8903 | 0.000312 | 99.6134 | 7.9908 | 8.1650 | 10,327.6722 | 0.0190 |
| (4, 5) | 257.9263 | 0.000402 | 99.6086 | 7.9930 | 8.1673 | 10,322.3450 | 0.0189 |
| (4, 6) | 245.1432 | 0.000211 | 99.6068 | 7.9888 | 8.1610 | 10,332.3070 | 0.0190 |
| (5, 7) | 246.8351 | 0.000288 | 99.6099 | 7.9910 | 8.1675 | 10,327.2332 | 0.0192 |
| (6, 8) | 251.8237 | 0.000292 | 99.6079 | 7.9838 | 8.1576 | 10,344.3522 | 0.0191 |
| (7, 11) | 265.3776 | 0.000062 | 99.6076 | 7.9850 | 8.1642 | 10,341.3786 | 0.0190 |
| (8, 10) | 246.0026 | 0.000325 | 99.6040 | 7.9899 | 8.1663 | 10,329.7822 | 0.0191 |
| (10, 12) | 270.9087 | 0.000475 | 99.6154 | 7.9929 | 8.1656 | 10,322.6447 | 0.0193 |
| (11, 13) | 248.0381 | 0.000261 | 99.6071 | 7.9907 | 8.1642 | 10,327.8064 | 0.0191 |
| (12, 13) | 275.6278 | 0.000286 | 99.6137 | 7.9891 | 8.1668 | 10,331.6947 | 0.0192 |
| (13, 15) | 265.7441 | 0.000124 | 99.6137 | 7.9858 | 8.1634 | 10,339.5298 | 0.0191 |
| (14, 16) | 256.9131 | 0.000133 | 99.6076 | 7.9919 | 8.1667 | 10,325.1165 | 0.0191 |
| (15, 17) | 250.9007 | 0.000385 | 99.6075 | 7.9923 | 8.1675 | 10,324.1188 | 0.0191 |
| (16, 17) | 284.1724 | 0.000190 | 99.6025 | 7.9937 | 8.1679 | 10,320.6825 | 0.0193 |
| (17, 19) | 242.0591 | 0.000175 | 99.6071 | 7.9909 | 8.1687 | 10,327.3806 | 0.0193 |
| (18, 20) | 266.2010 | 0.000142 | 99.6075 | 7.9881 | 8.1627 | 10,334.1432 | 0.0192 |
| (19, 21) | 258.7432 | 0.000328 | 99.6094 | 7.9898 | 8.1664 | 10,330.0628 | 0.0193 |
| (20, 22) | 243.6925 | 0.000065 | 99.6088 | 7.9886 | 8.1642 | 10,332.8555 | 0.0190 |
| (21, 23) | 278.0150 | 0.000333 | 99.6089 | 7.9916 | 8.1649 | 10,325.6493 | 0.0193 |
| (23, 25) | 258.9424 | 0.000051 | 99.6132 | 7.9895 | 8.1652 | 10,330.7675 | 0.0191 |
| (24, 26) | 259.5474 | 0.000279 | 99.6161 | 7.9901 | 8.1653 | 10,329.3328 | 0.0192 |
| (25, 27) | 261.0439 | 0.000176 | 99.6039 | 7.9938 | 8.1696 | 10,320.3900 | 0.0190 |
| (28, 30) | 275.7897 | 0.000157 | 99.6147 | 7.9920 | 8.1675 | 10,324.7755 | 0.0192 |
| (29, 31) | 256.4709 | 0.000029 | 99.6110 | 7.9910 | 8.1676 | 10,327.1585 | 0.0193 |
| (30, 32) | 262.2225 | 0.000239 | 99.6080 | 7.9915 | 8.1665 | 10,325.8447 | 0.0193 |
| (32, 36) | 292.9155 | 0.000005 | 99.6079 | 7.9874 | 8.1636 | 10,335.7033 | 0.0191 |
| (33, 35) | 266.9346 | 0.000153 | 99.6092 | 7.9933 | 8.1697 | 10,321.7698 | 0.0191 |
| (38, 45) | 231.5607 | 0.000285 | 99.6099 | 7.9888 | 8.1628 | 10,332.4847 | 0.0192 |

Our analysis demonstrates that Arnold's cat map parameters exhibit statistical independence, confirming that encryption quality is parameter-agnostic. The confidence intervals across all evaluated metrics show remarkable consistency, validating the robustness of our approach.

### *4.6 Performance analysis*

We evaluate the proposed cryptosystem for video encryption using mean squared error (MSE), peak signal-to-noise ratio (PSNR), weighted PSNR (wPSNR), and structural similarity index (SSIM) to assess its ability to obscure content and compared with AES.

**Table 10** Akiyo encryption metrics across different resolutions with 95% confidence intervals

| *Metric* | *Resolution* | *Mean* | *Std error* | *Lower CI* | *Upper CI* |
|---|---|---|---|---|---|
| Chi-square | 256 × 256 | 249.9207 | 12.7598 | 245.1562 | 254.6853 |
| | 512 × 512 | 254.8409 | 12.6692 | 250.1102 | 259.5717 |
| | 1,024 × 1,024 | 259.1769 | 13.4709 | 254.1468 | 264.2071 |
| Average correlation | 256 × 256 | 0.0011 | 0.0008 | 0.0009 | 0.0014 |
| | 512 × 512 | 0.000619 | 0.000339 | 0.000492 | 0.000745 |
| | 1,024 × 1,024 | 0.000236 | 0.000131 | 0.000187 | 0.000285 |
| NPCR | 256 × 256 | 99.6083 | 0.0152 | 99.6026 | 99.6139 |
| | 512 × 512 | 99.6095 | 0.0085 | 99.6063 | 99.6126 |
| | 1,024 × 1,024 | 99.6094 | 0.0034 | 99.6081 | 99.6106 |
| PSNR | 256 × 256 | 8.0001 | 0.0138 | 7.9950 | 8.0053 |
| | 512 × 512 | 7.9861 | 0.0064 | 7.9837 | 7.9885 |
| | 1,024 × 1,024 | 7.9902 | 0.0030 | 7.9891 | 7.9913 |
| wPSNR | 256 × 256 | 8.1747 | 0.0125 | 8.1701 | 8.1794 |
| | 512 × 512 | 8.1616 | 0.0065 | 8.1592 | 8.1640 |
| | 1,024 × 1,024 | 8.1656 | 0.0033 | 8.1644 | 8.1668 |
| MSE | 256 × 256 | 10,306.9449 | 32.7124 | 10,294.7299 | 10,319.1599 |
| | 512 × 512 | 10,338.8651 | 15.1826 | 10,333.1958 | 10,344.5343 |
| | 1,024 × 1,024 | 10,329.0968 | 7.1043 | 10,326.4441 | 10,331.7496 |
| SSIM | 256 × 256 | 0.0182 | 0.0011 | 0.0178 | 0.0186 |
| | 512 × 512 | 0.0188 | 0.0004 | 0.0187 | 0.0190 |
| | 1,024 × 1,024 | 0.0191 | 0.0001 | 0.0191 | 0.0192 |

MSE quantifies pixel-level differences between images, with higher values indicating greater distortion.

$$MSE = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [I(i,j) - K(i,j)]^2 \tag{13}$$

where $I(i,j)$, $K(i,j)$ are original and encrypted pixel values.

PSNR measures the ratio of an image's maximum signal strength to distorting noise, affecting its visual clarity. Higher PSNR values indicate better image quality with less distortion, while lower values reflect greater noise impact, which is ideal for encryption scenarios.

$$PSNR = 10 \cdot \log_{10} \left( \frac{255^2}{MSE} \right) (dB) \tag{14}$$

wPSNR is a vital quality metric that evaluates the weighted signal-to-noise ratio between the original image $O(x,y,z)$ and the encrypted image $E(x,y,z)$. Elevated wPSNR values suggest minimal visual distortion, whereas reduced values signify substantial distortion, suitable for robust encryption.

SSIM assesses image quality degradation due to processing, such as encryption, in this paper. It yields a value of 1 for identical images, with lower values indicating greater dissimilarity between the original and processed images.

**Table 11** Encryption metrics comparison

| *Video sequence* | *Method* | *PSNR (dB)* | *wPSNR (dB)* | *MSE* | *SSIM* |
|---|---|---|---|---|---|
| Akiyo | Our work | 7.9876 | 13.3194 | 10,335.1330 | 0.0188 |
| | AES | 7.9844 | 13.3112 | 10,342.7866 | 0.0186 |
| Bus | Our work | 7.6749 | 19.0731 | 11,106.7894 | 0.0152 |
| | AES | 7.6777 | 19.0911 | 11,099.5698 | 0.0152 |
| Carphone | Our work | 8.0577 | 12.9262 | 10,169.6424 | 0.0194 |
| | AES | 8.0516 | 12.9172 | 10,185.3334 | 0.0189 |
| Claire | Our work | 8.6062 | 11.9135 | 8,963.1117 | 0.0198 |
| | AES | 8.6004 | 11.9077 | 8,975.0355 | 0.0198 |
| Foreman | Our work | 7.7247 | 12.8257 | 10,980.2291 | 0.0202 |
| | AES | 7.7362 | 12.8335 | 10,951.2005 | 0.0212 |

The proposed block cipher effectively obscures multimedia content, as evidenced by the result shown in Table 11.

To further evaluate the efficiency of our proposed algorithm, we conducted a comparative analysis of the encryption and decryption speeds between our approach, which employs a confusion mechanism based on the Arnold's cat map, an algorithm utilising the Chirikov normal map function, and AES-CBC mode. We conducted the algorithm testing based on the following hardware specifications. The reported results were obtained using a desktop workstation (8-core CPU, 36GB RAM, Python and C++ optimised code), designed for algorithm benchmarking in a general-purpose environment. In Jiang et al. (2024), the Chirikov normal map requires numerous iterative lookups to retrieve values, resulting in extremely long decryption times on resource-constrained hardware. In contrast, the Arnold's cat map used in our work significantly reduces computation time by decreasing the number of iterations and optimising the access path (fewer iterations and lower memory-access overhead), making it more suitable for embedded environments.

**Table 12** Speed performance comparison

| *Algorithm* | *Encryption speed (Mb/s)* | *Decryption speed (Mb/s)* |
|---|---|---|
| Our work | 46.51 | 46.53 |
| Jiang et al. (2024) | 1.08 | 0.00041 |
| AES-CBC | 35.10 | 35.08 |

As shown in Table 12, the proposed algorithm achieves encryption speeds of 46.51 Mb/s and decryption speeds of 46.53 Mb/s, indicating relatively high performance in both phases. In contrast, the Chirikov normal map confusion records a significantly lower encryption speed of 1.08 Mbps and a decryption speed of 0.00041 Mbps. Our work demonstrates a better balance between encryption and decryption processes in terms of achieving robust security through the Arnold's cat map function while maintaining competitive encryption performance.

The measurements on the multiprocessing architecture into the confusion and diffusion processes corresponding to Algorithms 1 and 3 that average around 29 seconds and show minimal variation (≈0.06 seconds), demonstrating effective synchronisation between threads. This synchronisation enables the optimal utilisation of CPU resources, thereby reducing idle time and enhancing throughput. Proper thread coordination leads to improved performance, lower latency, and more efficient time management. Consequently, the system demonstrates superior performance and scalability, confirming the effectiveness of the parallel processing architecture in minimising execution time and maximising computational efficiency.

## 5 Conclusions

In conclusion, we present a chaos-based block cipher for video encryption that leverages chaotic maps to ensure comprehensive security while maintaining computational efficiency. Our approach implements a multiprocessing architecture that effectively secures video content across multiple implementations. The experiment result indicates that our encryption scheme, which utilises the Arnold's cat map for the confusion phase rather than inverse functions, achieves good performance compared to prior work across all evaluation metrics: even distribution in histograms, optimal NPCR (99.61408%) and UACI (33.45911%) values indicating strong key sensitivity, successful NIST statistical testing validation, and near-zero correlation coefficients eliminating pixel dependencies. Our algorithm achieves remarkable efficiency with encryption/decryption speeds of 46.51/46.53 Mbps, remaining competitive with AES-CBC. The multiprocessing architecture combined with the Arnold's cat map's two-dimensional mixing properties creates thorough pixel displacement with enhanced efficiency, successfully balancing security requirements with computational performance for video encryption applications. In the context of quantum attacks, our proposal still raises several concerns. First, similar to most conventional cryptographic schemes, it may potentially be vulnerable to quantum attacks if scalable quantum computers become available. Second, although we have validated the scheme on HD and 4K video streams, its scalability to 8K resolution or ultra-high frame rate applications necessitates further optimisation due to substantially increased computational and memory requirements. Looking ahead, we will expand our research in several directions. We plan to incorporate post-quantum secure primitives to enhance resistance against quantum adversaries and explore integration with homomorphic encryption to support the secure processing of encrypted multimedia streams. Additionally, we will evaluate the system on large-scale video datasets and in practical deployment scenarios, including streaming platforms and real-time medical imaging applications. These efforts will provide a more comprehensive understanding of the robustness, scalability, and applicability of our approach.

## Declarations

All authors declare that they have no conflicts of interest.

## References

Chen, B., Yu, S., Zhang, Z., Li, D. and Lü, J. (2021) 'Design and smartphone implementation of chaotic duplex h.264-codec video communications', *International Journal of Bifurcation and Chaos*, Vol. 31, No. 3, p.2150045.

Duan, L., Zhang, D., Xu, F. and Cui, G. (2018) 'A novel video encryption method based on faster R-CNN', in *Proceedings of the 2018 International Conference on Computer Science, Electronics and Communication Engineering (CSECE 2018)*, Atlantis Press, pp.100–104.

El-Shafai, W., Almomani, I.M. and Alkhayer, A. (2021) 'Optical bit-plane-based 3D-JST cryptography algorithm with cascaded 2D-FrFT encryption for efficient and secure HEVC communication', *IEEE Access*, Vol. 9, pp.35004–35026.

Elkamchouchi, H., Anton, R. and Abouelseoud, Y. (2022) 'Multimedia data secure transmission: a review', *International Journal of Scientific Research and Management (IJSRM)*, Vol. 10, No. 12, pp.949–971.

Faragallah, O., Sallam, A., Alajmi, M. and El-Sayed, H. (2022) 'Low-latency selective encryption scheme for scalable extension of high-efficiency video coding', *Transactions on Emerging Telecommunications Technologies*, Vol. 33, No. 11, p.e4599.

He, J. (2023) 'Chaotic sequence-based video encryption algorithm for network surveillance', *Journal of Network Intelligence*, Vol. 8, No. 3, pp.676–692.

Hosny, K.M., Zaki, M.A., Lashin, N.A., Fouda, M.M. and Hamza, H.M. (2023) 'Multimedia security using encryption: a survey', *IEEE Access*, Vol. 11, pp.63027–63056.

Ibrahem, M.K. and Qasim, H. (2021) 'VoIP speech encryption system using stream cipher with chaotic key generator', *Iraqi Journal of Science*, Special Issue, pp.240–248.

Jiang, D., Chen, T., Yuan, Z., Li, W-X., Wang, H-T. and Lu, L-L. (2024) 'Real-time chaotic video encryption based on multi-threaded parallel confusion and diffusion', *Information Sciences*, Vol. 666, p.120420.

Lin, Z., Feng, Y. and Liang, S. (2023) 'Cryptanalysis of a multiround image encryption algorithm based on 6D self-synchronizing chaotic stream cipher', *International Journal of Bifurcation and Chaos*, Vol. 33, No. 3, p.2350028.

Liu, F. and Koenig, H. (2010) 'A survey of video encryption algorithms', *Computers & Security*, Vol. 29, No. 1, pp.3–15.

Maolood, A., Gbashi, E. and Shakir, E. (2022) 'Novel lightweight video encryption method based on ChaCha20 stream cipher and hybrid chaotic map', *International Journal of Electrical and Computer Engineering (IJECE)*, Vol. 12, No. 5, pp.4988–5000.

Obaida, T.H., Jamil, A.S. and Hassan, N.F. (2022) 'Improvement of rabbit lightweight stream cipher for image encryption using Lévy flight', *International Journal of Health Sciences*, Vol. 6, No. S8, pp.1628–1641.

Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S., Levenson, M., Vangel, M., Banks, D., Heckert, A., Dray, J. and Vo, S. (2010) *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*, Technical Report NIST SP 800-22 Rev. 1a, National Institute of Standards and Technology.

Shah, R.A., Asghar, M.N., Abdullah, S., Kanwal, N. and Fleury, M. (2020) 'SLEPX: an efficient lightweight cipher for visual protection of scalable HEVC extension', *IEEE Access*, Vol. 8, pp.187784–187807.

Zia, U., McCartney, M., Scotney, B. et al. (2022) 'Survey on image encryption techniques using chaotic maps in spatial, transform and spatiotemporal domains', *International Journal of Information Security*, Vol. 21, No. 4, pp.917–935.