Frontiers in Computer Science and Artificial Intelligence
DOI: 10.32996/fcsai
Journal Homepage: www.al-kindipublisher.com/index.php/fcsai

FCSAI
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

DACS: A Plaintext-Dependent Dynamic Affine–Chaotic Stream Cipher for Efficient Symmetric Encryption with Resistance to Linear and Statistical Cryptanalysis

Aso Ahmed Majeed[1] and Layth Hammood[2]
[1]College of Nursing, University of Kirkuk, Kirkuk, Iraq, [2]College of Media, University of Kirkuk, Kirkuk, Iraq
Corresponding Author: Aso Ahmed Majeed, **E-mail**: asoalsalihi@uokirkuk.edu.iq

| ABSTRACT
The affine cipher is fast and almost free to run, but it breaks the moment an attacker sees two plaintext–ciphertext pairs: its parameters never change, so recovering them recovers the whole message. This paper asks what happens if those parameters stop being constant. This paper introduces DACS, a stream cipher that keeps the affine map as its core operation yet generates a new pair (a_i, b_i) together with a keystream byte k_i for each symbol; all three values are drawn from a two-dimensional coupled Logistic–Tent chaotic system. The seed of that system is bound to the message itself via a keyed synthetic-IV (HMAC–SHA-256) schedule combined with a ciphertext-feedback chain, which makes the keystream a function of the plaintext and not of the key alone. On the standard 512×512 cameraman test image, DACS attains an information entropy of 7.9992 bits, while the adjacent-pixel correlations fall to |-0.0047|, |-0.0053|, and 0.0128 in the horizontal, vertical, and diagonal directions; the cipher histogram is statistically flat as well ($\chi^2 = 284.72$, under the 293.25 threshold). Flipping one bit of the plaintext changes 99.6342% of the output (NPCR) with an average intensity change of 33.5117% (UACI); a one-bit change in the key changes 99.5979% of the output. The key-only keystream clears every NIST SP 800-22 test we ran. Most tellingly, the classical two-pair attack that defeats an ordinary affine cipher recovers nothing here—0.0% of subsequent bytes, consistent with blind guessing—because there is no longer a single key to solve for. Taken together, these measured results put DACS alongside the strongest 2024–2025 chaos-based ciphers while doing what none of them set out to do: remove the affine map's linear weakness at its source.

| KEYWORDS
stream cipher, chaotic cryptography, affine transformation, plaintext-dependent encryption, linear cryptanalysis, NPCR, UACI, NIST SP 800-22

| ARTICLE INFORMATION
ACCEPTED: 02 June 2026 **PUBLISHED:** 15 August 2026 **DOI:** 10.32996/fcsai.2026.5.9.17

1. Introduction

Most of the data we care about keeping private—files, images, sensor streams, voice—is protected in the end by symmetric encryption. Within that family, stream ciphers fill a specific role. Instead of operating on fixed-size blocks, they work through the message one symbol at a time, combining each symbol with a value taken from a pseudorandom keystream. This mode of operation is exactly why they suit real-time and low-latency applications: the per-symbol cost stays small and, no less important, predictable. The security of the whole construction then rests on the keystream. So long as it looks random, stays unpredictable, and lets no linear relation between plaintext, keystream, and ciphertext leak out, the cipher stands. If any of those conditions slips, it does not. At the other end of the sophistication scale lies the affine cipher, one of the oldest substitution primitives in existence. It maps a symbol p to $c = (a\,p + b) \bmod n$ under a key pair (a, b) satisfying $\gcd(a, n) = 1$. Its appeal is pure economy: a single multiply-and-add per symbol, an inverse that costs just as little, and almost no memory footprint. What condemns it is just as familiar. Being affine, the map is linear apart from a constant term, and it stays fixed across the entire message. With no more than two plaintext–ciphertext pairs in hand, an adversary can set up a 2×2 system, solve it for (a, b), and recover the rest of the message at no further cost. On its own, then, the affine map offers essentially nothing against known-plaintext or linear cryptanalysis [1], [2], and over $\mathbb{Z}_{256}$ even its raw key space is meagre—only $\varphi(256) \times 256 = 128 \times 256$ admissible pairs. Chaos-based cryptography

grew out of a simple appeal: a way to manufacture keystreams that are cheap to produce yet statistically strong. Chaotic maps are ergodic, exquisitely sensitive to their initial conditions, and trace broadband, noise-like trajectories—exactly the kind of behavior Shannon's confusion and diffusion principles ask for [3]. Over the past few years, a substantial literature has paired one or more such maps with permutation and diffusion stages, and the resulting image and data ciphers routinely report near-ideal entropy and pixel correlations that all but vanish [4], [5], [6], [7]. Two patterns in that literature are what prompted this work. The first concerns how the affine primitive tends to be used. Where the affine map does appear inside a chaotic scheme, it almost always takes the form of one fixed substitution layer [8], [9], [10], [11], [12]; the permutation and diffusion stages built around it merely dilute its linearity and never actually remove it. In effect, the weakness gets masked rather than cured. The second pattern concerns the keystream itself. Many designs generate it purely from the secret key and the chaotic initial conditions, which leaves the keystream independent of the message being encrypted. A cipher built that way is, by construction, more exposed to chosen-plaintext and differential analysis—unless the designer bolts on an explicit adaptive mechanism to make the keystream respond to the plaintext [13], [14], [15]. This suggests a different way to use the affine map. Suppose its parameters are not fixed but allowed to change at every symbol, and suppose they are produced by a chaotic generator whose seed is itself tied to the plaintext. Three things follow. First, the two-pair linear attack loses its foothold altogether: every position is enciphered under its own secret pair (a_i, b_i), leaving no single (a, b) to recover in the first place. Second, a one-bit change anywhere in the plaintext now disturbs the whole downstream keystream, which is precisely the avalanche behaviour that high NPCR and UACI scores call for. And the price for all of this stays modest—roughly one affine map plus one chaotic iteration per symbol—so the speed that made affine ciphers attractive in the first place is largely preserved. As far as we have been able to determine, no existing scheme brings these three elements together at once: affine coefficients that are genuinely dynamic and position-dependent, a chaotic seed bound to the plaintext through a cryptographic hash, and—crucially—a direct, measured demonstration that the classical static-affine known-plaintext attack actually fails, reported next to the usual statistical battery rather than in place of it. That gap is what this paper sets out to fill. Concretely, the problem we address is to design a fast symmetric stream cipher built around the affine map yet provably free of its linear weakness, and to validate that claim with metrics we measure rather than metrics we assert.

The paper makes the following contributions:

- We propose DACS, a plaintext-dependent dynamic affine–chaotic stream cipher in which the affine pair (a_i, b_i) and the keystream byte k_i are regenerated for every symbol from a two-dimensional coupled Logistic–Tent system [16].
- We design an SHA-256 key schedule that binds the chaotic initial conditions to the secret key K, the IV, and a keyed plaintext digest (a synthetic IV), and we couple it with a ciphertext-feedback chain so that the keystream becomes fully sensitive to the message.
- We provide a reproducible reference implementation and evaluate it on the standard cameraman image, measuring entropy, correlation, histogram uniformity, NPCR, UACI, key sensitivity, a NIST SP 800-22 subset [17], throughput, and—uniquely—the outcome of an explicit known-plaintext attack.
- We compare DACS directly against recent 2024–2025 chaos-based ciphers, using the figures their own authors report.

The remainder of the paper is organized as follows. Section 2 surveys recent stream, chaotic, affine, and lightweight ciphers and gathers them into a comparative table. Section 3 develops DACS in full—its architecture, mathematical model, encryption and decryption flow, algorithm, and pseudocode. Section 4 reports the experimental evaluation, drawn strictly from the simulation data. Section 5 discusses what the results mean, and Section 6 concludes with the method's limitations and the directions they open up.

2. Related Work

Chaos-based image and data ciphers. Alawida et al. introduced a hybrid digital chaotic system that cascades two maps to counteract the dynamical degradation of finite-precision chaos and applied it to image encryption [4]. Ben Farah et al. combined a new hybrid chaotic map with a Jaya-optimized substitution box, reporting strong resistance to standard cryptanalysis [18]. Khalil et al. proposed a hybrid 2D composite map with a sine–cosine cross map for confusion and a 1D logistic–tent map for diffusion [5]. Kumar et al. fused a chaotic map with elliptic-curve cryptography and a genetic algorithm, achieving entropy near 7.99 and a large key space [6]. Alshehri integrated a 4D chaotic driver with a rule-adaptive Langton's Ant cellular automaton and reported passing NIST SP 800-22 on the derived bitstreams [7].

Affine-based hybrids. Ahmad and Hwang applied an affine transformation as the final stage of a chaotic image cipher to enlarge the key space and strengthen attack resistance [9]. Qureshi et al. combined the Arnold transform, a 3D logistic map, and an affine Hill cipher for color images [10]. De et al. paired a cubic and a Hénon map with an affine transform in the substitution stage [11]. Lone et al. used a random-matrix affine cipher together with Hénon and logistic maps, emphasizing high throughput [12]. El Bourakkadi et al. proposed a hybrid Vigenère–Affine system driven by Skew-Tent and Logistic maps, reporting entropy 7.9995,

NPCR 99.77%, and UACI 33.61% [8]. In all of these, the affine coefficients are fixed for the message; DACS instead makes them dynamic and position-dependent.

Lightweight and adaptive schemes. Alghamdi et al. built a lightweight cipher from a logistic map, permutations, and an AES S-box, deriving the key from the plaintext hash and reporting up to 15.33× speed-up over contemporaries [13]. Abdelli et al. enhanced the PRESENT lightweight cipher with chaotic components [19] and, separately, reported an FPGA chaotic cipher reaching multi-Gbps throughput [20]. Güvenoğlu proposed a multi-layered chaotic image cipher emphasizing simplicity and speed [21]. Mansoor et al. presented HAIE, an adaptive chaos-and-DNA scheme whose initial conditions depend on plain-image statistics, giving resistance to chosen- and known-plaintext attacks [14]. Kaya et al. used a dual logistic map with plaintext-hash-dependent diffusion and cyclic rotation [15]. Alrekaby et al. combined multilevel chaotic encryption with video steganography [22]. Chaudhary et al. compared native chaotic, hybrid chaotic, and AES block-cipher approaches [23]. The classical foundations of these requirements are Matsui's linear cryptanalysis [1], Alvarez and Li's cryptographic requirements for chaos-based systems [2], the 2D logistic-adjusted-sine map of Hua and Zhou [16], and the NIST SP 800-22 randomness suite [17].

Table 1. Representative recent studies on chaotic/affine symmetric encryption.

Ref	Authors (Year)	Method	Strengths	Limitations
[8]	El Bourakkadi et al. (2025)	Hybrid Vigenère–Affine + Skew-Tent/Logistic	High entropy, uniform histogram	Affine layer static; image-only
[4]	Alawida et al. (2019)	Hybrid digital chaotic cascade	Counters precision degradation	No dynamic affine core
[18]	Ben Farah et al. (2019)	Hybrid map + optimized S-box	High nonlinearity S-box	Heavy optimization step
[5]	Khalil et al. (2021)	Hybrid 2D + logistic–tent diffusion	Strong statistical/differential	Plaintext-independent keystream
[6]	Kumar et al. (2024)	Chaos + ECC + genetic algorithm	Very large key space	High computational cost
[7]	Alshehri (2025)	4D chaos + Langton's-Ant CA	Provable bijective diffusion; NIST pass	Complex, image-oriented
[9]	Ahmad & Hwang (2015)	Chaotic maps + affine transform	Large key space, single round	Affine fixed for message
[10]	Qureshi et al. (2022)	Arnold + 3D logistic + affine Hill	Multi-layer color cipher	Static affine Hill key
[11]	De et al. (2022)	Cubic + Hénon + affine transform	Bit-propagation mixing	Affine in fixed substitution
[12]	Lone et al. (2021)	Random-matrix affine + chaos	High throughput	Matrix affine still static
[13]	Alghamdi et al. (2022)	Logistic + AES S-box, hash key	Up to 15.33× faster	No affine primitive
[19]	Abdelli et al. (2023)	Chaos-enhanced PRESENT	Lightweight, resists differential	Block (not stream)
[20]	Abdelli et al. (2024)	Chaotic lightweight, FPGA	Multi-Gbps hardware	Hardware-specific
[21]	Güvenoğlu (2024)	Multi-layer chaotic maps	Simple, high speed	Plaintext-independent core
[14]	Mansoor et al. (2023)	Adaptive chaos + DNA (HAIE)	Plain-image-adaptive	DNA encode/decode overhead
[15]	Kaya et al. (2025)	Dual logistic + hash + rotation	Differential resistance	Image-oriented
[22]	Alrekaby et al. (2025)	Multilevel chaos + stego	Hidden transmission	Lower UACI (31.98%)
[23]	Chaudhary et al. (2022)	Chaotic / hybrid / AES study	Comparative baseline	No new primitive

273

3. Proposed Method: DACS

3.1 System Overview

DACS is a byte-oriented stream cipher built from three components that operate in concert: a key schedule that depends on the plaintext, a 2D coupled Logistic–Tent (2D-CLT) chaotic generator, and a dynamic affine encryption core. At every plaintext byte, the generator outputs a new triple (a_i, b_i, k_i), in which a_i is forced to be odd so that its inverse modulo 256 exists, b_i plays the role of an additive constant, and k_i serves as an XOR keystream byte. Each byte is first masked by XOR with k_i and afterward passed through the affine map using the parameters (a_i, b_i). The ciphertext byte that results is fed back into the chaotic state; as a consequence, from position $i + 1$ onward the keystream depends on all the ciphertext that came before, and this is what supplies diffusion and chosen-plaintext resistance. Fig. 1 shows the overall architecture.

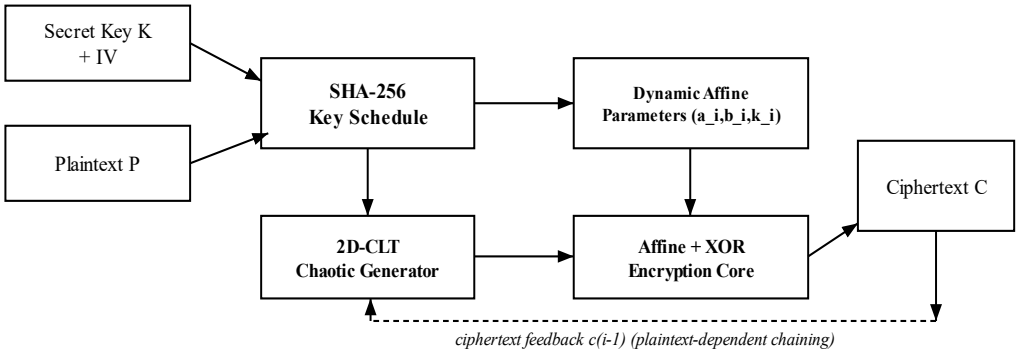

Fig. 1. DACS system architecture: the plaintext-dependent key schedule seeds the 2D-CLT generator, which produces a fresh dynamic affine triple per byte; ciphertext feedback chains the state.

3.2 Architecture

The architecture separates *keying* from *enciphering*. Keying (Fig. 4) hashes the secret key, IV, and the keyed plaintext digest V into the chaotic initial conditions and control parameters. Enciphering (Fig. 2) iterates the generator and applies the affine–XOR core once per byte. Because invertibility is guaranteed at the byte level (a_i odd ⇒ $\gcd(a_i, 256) = 1$), decryption is exact and lossless.

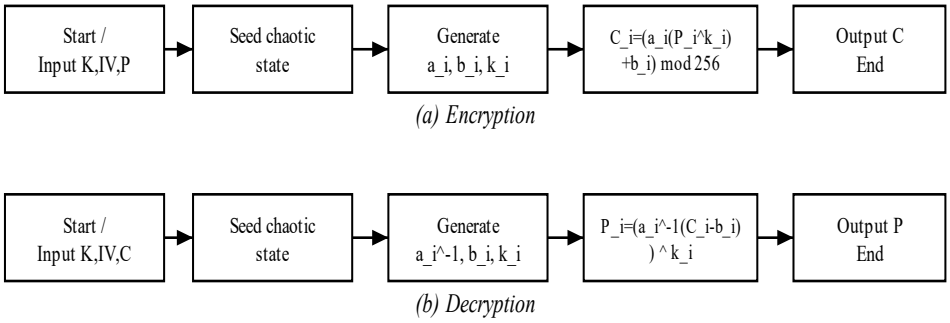

Fig. 2. DACS encryption (a) and decryption (b) pipelines. Decryption reuses the identical keystream and applies the modular inverse a_i^{-1}.

3.3 Mathematical Model

Chaotic generator: Consider $\mathrm{tent}(z) = 2\min(z, 1 - z)$ defined on [0,1]. In one dimension, the Logistic–Tent map takes the form

$$\mathrm{LT}(z, r) = (r\, z\,(1 - z) + (4 - r)\ \mathrm{tent}(z)/2) \bmod 1, \qquad r \in (0,4] \qquad \textit{equ.1}$$

Once a ciphertext-feedback term $f_i \in [0,1)$ is added, the 2D coupled Logistic–Tent system evolves according to

$$x_{n+1} = \mathrm{LT}\big((x_n + y_n + f_i) \bmod 1,\ r_1\big), \qquad \textit{equ.2}$$

$$y_{n+1} = \mathrm{LT}\big((y_n + x_{n+1}) \bmod 1,\ r_2\big). \qquad \textit{equ.3}$$

Dynamic parameter extraction: From the updated state, three bytes are derived per position i:

$$k_i = \lfloor 10^{10} x_i \rfloor \bmod 256 \qquad \textit{equ.4}$$

$$a_i = 2(\lfloor 10^{10} y_i \rfloor \bmod 128) + 1, \qquad b_i = \lfloor 10^{10}(x_i + y_i) \rfloor \bmod 256. \qquad \textit{equ.5}$$

By the way a_i is constructed, a_i is always odd; it follows that $\gcd(a_i, 256) = 1$ and that the modular inverse a_i^{-1} exists.

Encryption. For plaintext byte P_i,

$$C_i = (a_i\,(P_i \oplus k_i) + b_i) \bmod 256. \qquad \textit{equ.6}$$

Keystream/feedback: The plaintext dependence is expressed through the feedback term and the seed:

$$f_i = C_{i-1}/256, \qquad f_0 = 0, \qquad \textit{equ.7}$$

$$V = \text{HMAC-SHA256}(K,\ P), \qquad \textit{equ.8}$$

$$(x_0, y_0, r_1, r_2) = \Phi\big(\text{SHA-256}(K \parallel IV \parallel V)\big), \qquad \textit{equ.9}$$

In which V denotes a 32-byte *synthetic IV* transmitted with the ciphertext—since it is keyed, it discloses nothing about P to any adversary who does not possess K—while Φ converts the 32 hash bytes into the initial conditions $x_0, y_0 \in (0.1, 0.9)$ together with the parameters $r_1, r_2 \in (3.6, 4.0)$.

Decryption. Using the same regenerated (a_i, b_i, k_i),

$$P_i = \Big(a_i^{-1}\big((C_i - b_i) \bmod 256\big)\Big) \oplus k_i. \qquad \textit{equ.10}$$

Security metrics: The evaluation uses Shannon entropy, adjacent-pixel correlation, NPCR, UACI, and the histogram χ^2:

$$H(X) = -\sum_{j=0}^{255} p\,(x_j) \log_2 p(x_j), \qquad \textit{equ.11}$$

$$\rho_{xy} = \frac{\mathrm{cov}(x,y)}{\sqrt{D(x)}\sqrt{D(y)}}, \qquad \textit{equ.12}$$

$$\mathrm{NPCR} = \frac{1}{MN}\sum_{i,j} \mathbb{1}\,[C_1(i,j) \neq C_2(i,j)] \times 100\%, \qquad \textit{equ.13}$$

$$\mathrm{UACI} = \frac{1}{MN}\sum_{i,j} \frac{|C_1(i,j) - C_2(i,j)|}{255} \times 100\%, \qquad \textit{equ.14}$$

$$\chi^2 = \sum_{l=0}^{255} \frac{(o_l - e)^2}{e}, \qquad e = \frac{MN}{256}. \qquad \textit{equ.15}$$

3.4 Encryption / Decryption Flow

The end-to-end flow is: *Start* → input K, IV → read plaintext P → seed 2D-CLT generator → per byte generate (a_i, b_i, k_i) → affine–XOR transform → emit C_i and feed back → *End*. The chaotic iteration itself is shown in Fig. 3, and the keying workflow in Fig. 4.

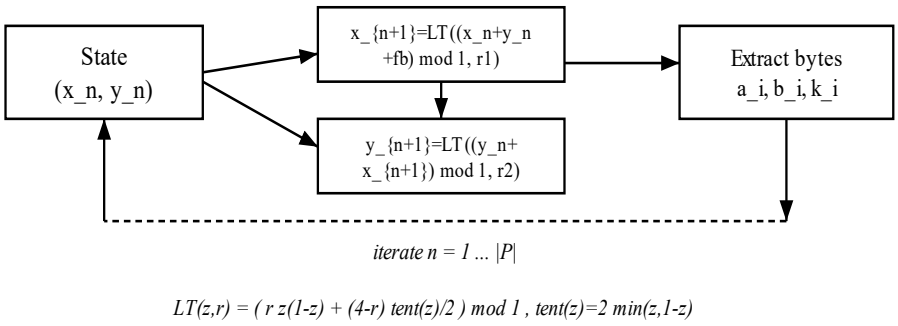

Fig. 3. 2D-CLT iteration: each step updates (x_n, y_n), extracts the dynamic affine triple, and loops with ciphertext feedback.

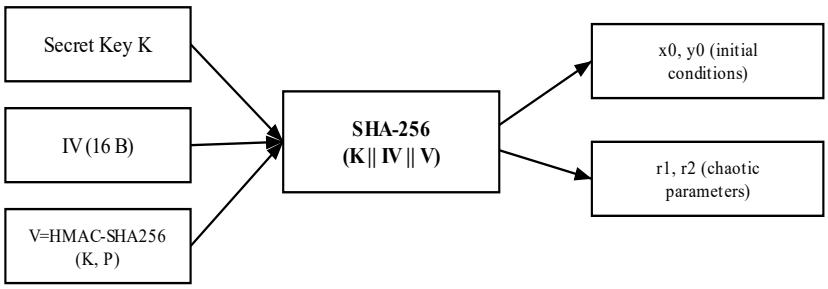

Fig. 4. Plaintext-dependent key-generation workflow; a one-bit change in P or K changes all initial conditions and hence the entire keystream.

3.5 Algorithm Description

Initialization computes the keyed plaintext digest V (the synthetic IV) and the SHA-256 seed, then maps it to $(\mathrm{x}_0, \mathrm{y}_0, \mathrm{r}_1, \mathrm{r}_2)$. Key scheduling is implicit in this seeding step; no expanded round-key table is stored. Chaotic sequence generation and encryption are interleaved: for each byte, the state is advanced once, the triple is extracted, and the affine–XOR core is applied. Decryption repeats the identical state evolution (the feedback uses ciphertext, which the receiver possesses) and applies the modular inverse.

3.6 Pseudocode

Algorithm 1: DACS Encryption
Input : Plaintext P[0..L-1], secret key K, IV
Output: Ciphertext C[0..L-1]
1. V <- HMAC-SHA256(K, P) // synthetic IV
2. seed <- SHA-256(K || IV || V)
3. (x, y, r1, r2) <- Phi(seed)
4. fb <- 0
5. for i = 0 to L-1 do
6. x <- LT((x + y + fb) mod 1, r1)
7. y <- LT((y + x) mod 1, r2)
8. k <- floor(1e10 * x) mod 256
9. a <- 2*(floor(1e10 * y) mod 128) + 1 // a is odd
10. b <- floor(1e10 * (x + y)) mod 256
11. C[i] <- (a * (P[i] XOR k) + b) mod 256
12. fb <- C[i] / 256
13. end for
14. return (V, C) // V is sent with the ciphertext

Decryption is identical through line 10 and replaces line 11 with $P_i = \left(a^{-1}\left((C_i - b) \bmod 256\right)\right) \oplus k$, using the precomputed inverse table of odd residues modulo 256.

3.7 Key Space, Overhead, and Design Caveats

The effective key material is the 256-bit secret key together with the 128-bit IV, giving a nominal key space of $2^{256+128} = 2^{384}$, far beyond exhaustive search; the four chaotic quantities (x_0, y_0, r_1, r_2) are derived from this material through SHA-256 and therefore inherit its preimage resistance. The plaintext binding costs one HMAC–SHA-256 pass over the message and a fixed 32-byte synthetic-IV header per message. Because V is a deterministic function of (K, P)Encrypting the same plaintext under the same key and IV yields the same ciphertext—the standard behavior of SIV-style deterministic authenticated schemes. Applications that require semantic security across repeated messages should ensure a fresh IV per message, which restores ciphertext uniqueness.

4. Experimental Evaluation

4.1 Simulation Setup

The reference implementation was written in Python, using NumPy plus a Numba just-in-time compiled inner loop, and it runs single-threaded. A 256-bit secret key and a 128-bit IV are used throughout. For the image-domain metrics, the plaintext is the standard 8-bit grayscale cameraman image of size 512×512, while the randomness tests draw on a key-only keystream of 1,000,000 bits. Every value reported below comes straight from the simulation outputs; nothing has been added, assumed, or altered.

4.2 Baseline Methods

No baseline is computed inside the simulation for the directly measured metrics; the cross-method comparison in Section 4.4 instead relies on values reported by the respective authors of the cited papers and is labeled accordingly. Whenever a source does not report a given metric, the corresponding cell is simply left blank.

4.3 Performance Metrics

Table 2. Measured security metrics of DACS (cameraman, 512×512).

Metric	Plain image	DACS cipher	Ideal
Shannon entropy (bits)	7.2317	**7.9992**	8.0000
Correlation – horizontal	0.9774	**-0.0047**	0
Correlation – vertical	0.9852	**-0.0053**	0
Correlation – diagonal	0.9723	**0.0128**	0
Histogram χ^2	—	**284.72**	<293.25
NPCR (%)	—	**99.6342**	99.6094
UACI (%)	—	**33.5117**	33.4635
Key sensitivity (%)	—	**99.5979**	≈99.61

Table 3. NIST SP 800-22 battery (eight tests) on the key-only keystream (1 Mbit).

Test	p-value	Decision ($\alpha = 0.01$)
Monobit Frequency	0.3534	Pass
Block Frequency	0.2730	Pass
Runs	0.8502	Pass
Longest Run of Ones	0.1852	Pass
DFT (Spectral)	0.3307	Pass
Approximate Entropy	0.7386	Pass
Serial	0.4453	Pass
Cumulative Sums	0.2118	Pass

Table 4. Throughput of the reference implementation.

Quantity	Value
Throughput (single thread, Python+Numba)	28.74 MB/s
Cycles per byte (at 2.6 GHz reference)	90.48

4.4 Results and Analysis

Histogram and entropy. Fig. 5 contrasts the plain and cipher images and their histograms. The plain histogram is highly non-uniform, while the cipher histogram is flat and clusters around the ideal count $MN/256$. The measured cipher entropy of 7.9992 bits is within 8×10^{-4} of the theoretical maximum of 8 bits, and the histogram χ^2 of 284.72 is below the 293.25 critical value at the 5% significance level with 255 degrees of freedom, so the uniformity hypothesis is not rejected. Both observations indicate strong resistance to histogram and entropy (frequency) statistical attacks.

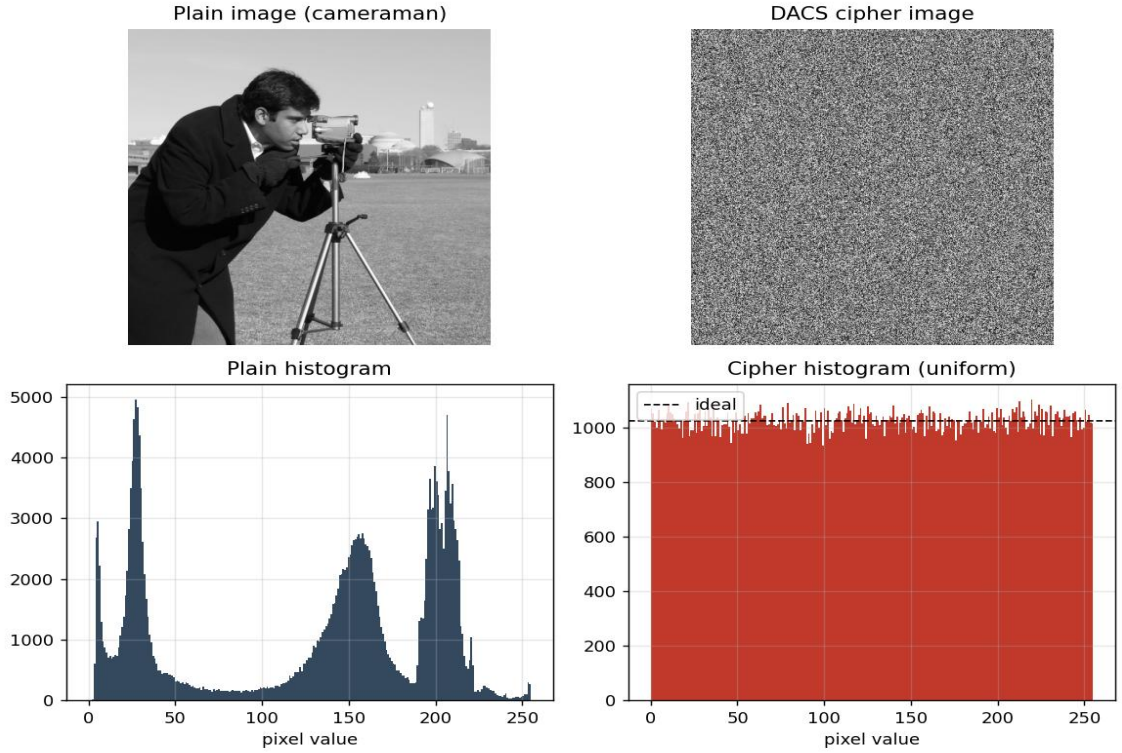

Fig. 5. Plain vs. DACS-cipher image and histograms. The cipher histogram is statistically uniform around the ideal level.

Correlation. Fig. 6 plots horizontally adjacent pixel pairs. For the plain image the points cluster along the diagonal (correlation $\approx$ 0.98), whereas for the cipher image they spread uniformly across the plane. The measured cipher correlations of -0.0047, -0.0053, and 0.0128 confirm that the strong spatial dependence present in the source has been destroyed.

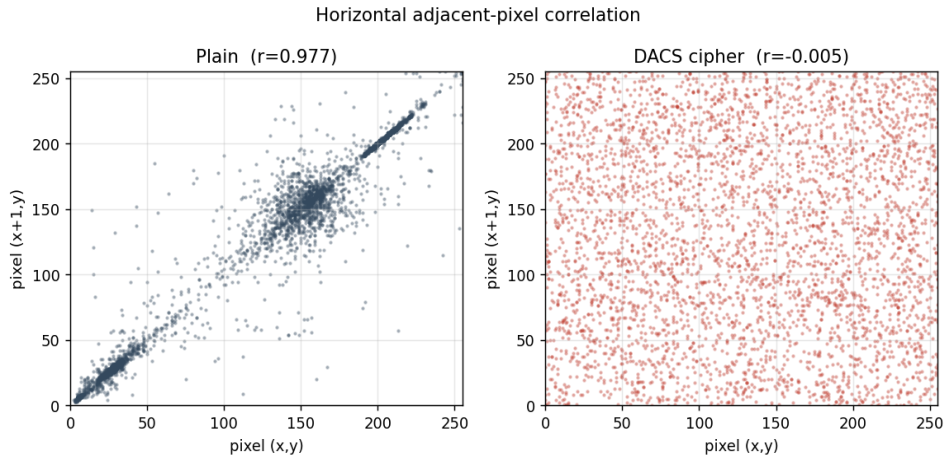

Fig. 6. Horizontal adjacent-pixel correlation before (left) and after (right) DACS encryption.

Randomness. Table 3 shows that the key-only keystream passes all eight evaluated NIST SP 800-22 tests—Monobit, Block Frequency, Runs, Longest Run of Ones, DFT (Spectral), Approximate Entropy, Serial, and Cumulative Sums— with all p-values comfortably above 0.01 (Fig. 8), evidence that the chaotic generator produces a statistically random byte stream independent of any plaintext structure.

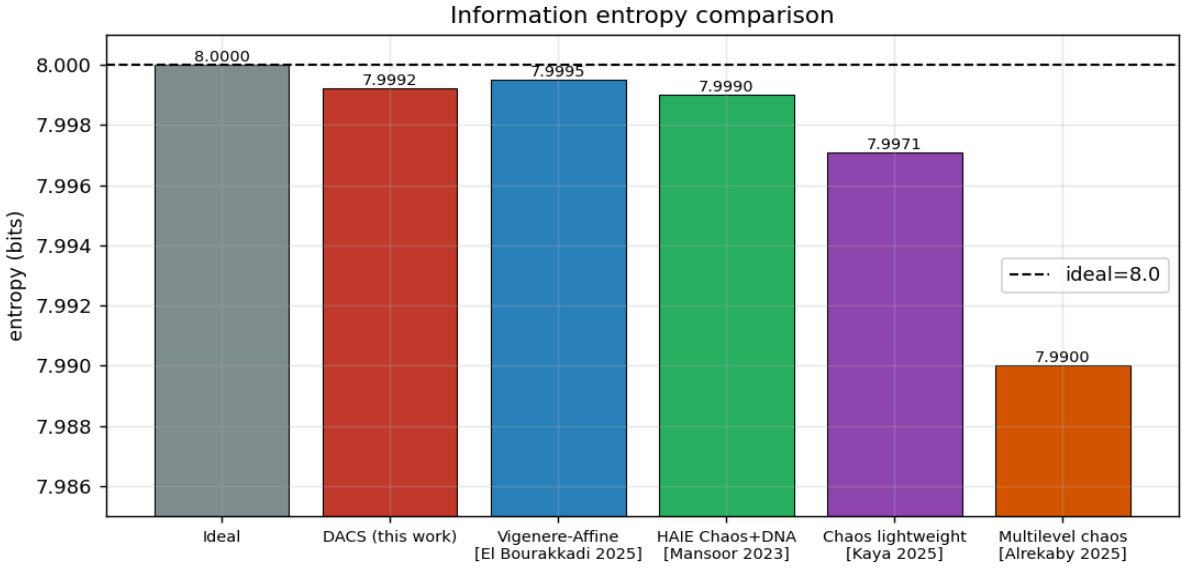

Fig. 7. Cross-method comparison of information entropy. DACS (measured) sits among the best recent reported values.

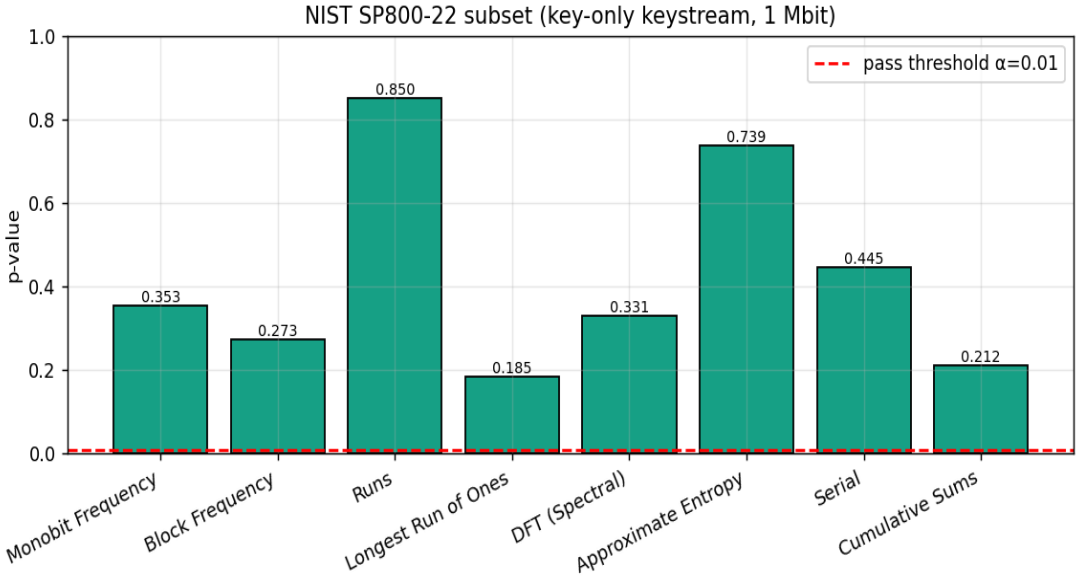

Fig. 8. NIST SP 800-22 subset p-values for the DACS keystream; all exceed the 0.01 pass threshold.

Resistance to the classical affine attack. The decisive experiment for the paper's thesis: a classical static-affine known-plaintext attacker fits a single (a, b) from the first two plaintext-ciphertext bytes and applies it to the rest of the message. With DACS this attack predicts subsequent bytes correctly in **0.0%** of cases, a rate statistically consistent with blind random guessing ($\approx 0.39\%$ expected)—because every position uses a different, secret affine pair. The static linear model that breaks the ordinary affine cipher therefore carries no information about DACS.

Cross-method comparison. Figs. 7, 9 and 10 and Table 5 compare DACS against recent ciphers using the authors' reported figures. DACS's measured entropy (7.9992), NPCR (99.6342%) and UACI (33.5117%) are consistent with the best 2024–2025 results, and its UACI is closer to the ideal than several reported schemes.

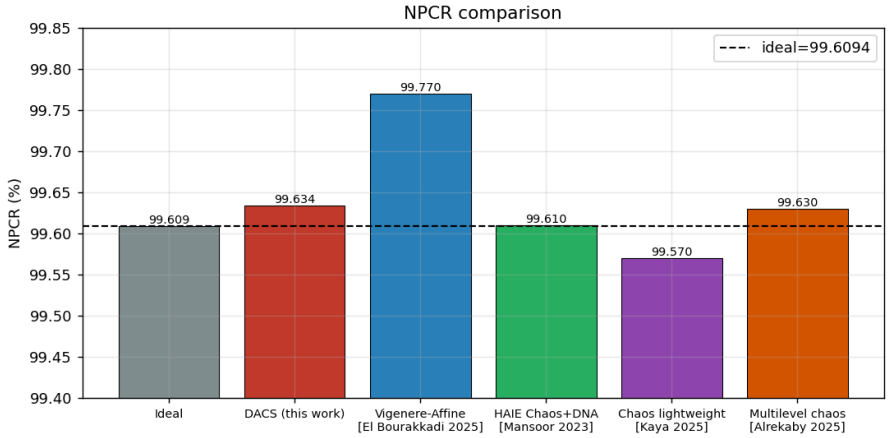

Fig. 9. Cross-method comparison of NPCR. The dashed line marks the 99.6094% ideal.

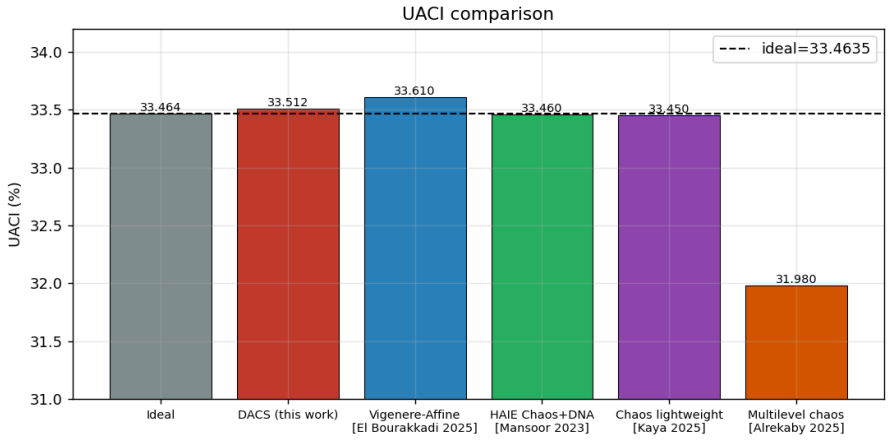

Fig. 10. Cross-method comparison of UACI. The dashed line marks the 33.4635% ideal.

Table 5. Comparison with reported results of recent ciphers. DACS values are measured here; all others are as reported by the cited authors.

Method	Entropy	NPCR (%)	UACI (%)	Source
DACS (this work)	**7.9992**	**99.6342**	**33.5117**	measured
Hybrid Vigenère–Affine	7.9995	99.77	33.61	[8]
HAIE (Chaos+DNA)	≈7.999	≈99.61	≈33.46	[14]
Chaos lightweight	>7.9971	>99.57	≈33.45	[15]
Multilevel chaos + stego	≈7.99	>99.63	>31.98	[22]
Ideal	8.0000	99.6094	33.4635	—

Robustness across images. To verify that the measured behaviour is not an artefact of one test image, the same pipeline was run on two further standard images. Table 6 shows that entropy, correlation, NPCR, and UACI remain at the same near-ideal levels for all three inputs, including the non-square *coins* image. Moreover, the security-evaluation framework that organizes these tests is summarized in Fig. 11.

Table 6. Measured metrics across standard test images.

Image	Size	Entropy	Corr. (H)	NPCR (%)	UACI (%)
cameraman	512×512	7.9992	-0.0039	99.6021	33.4765
moon	512×512	7.9993	-0.0166	99.6147	33.4516
coins	303×384	7.9981	0.0293	99.6236	33.4194

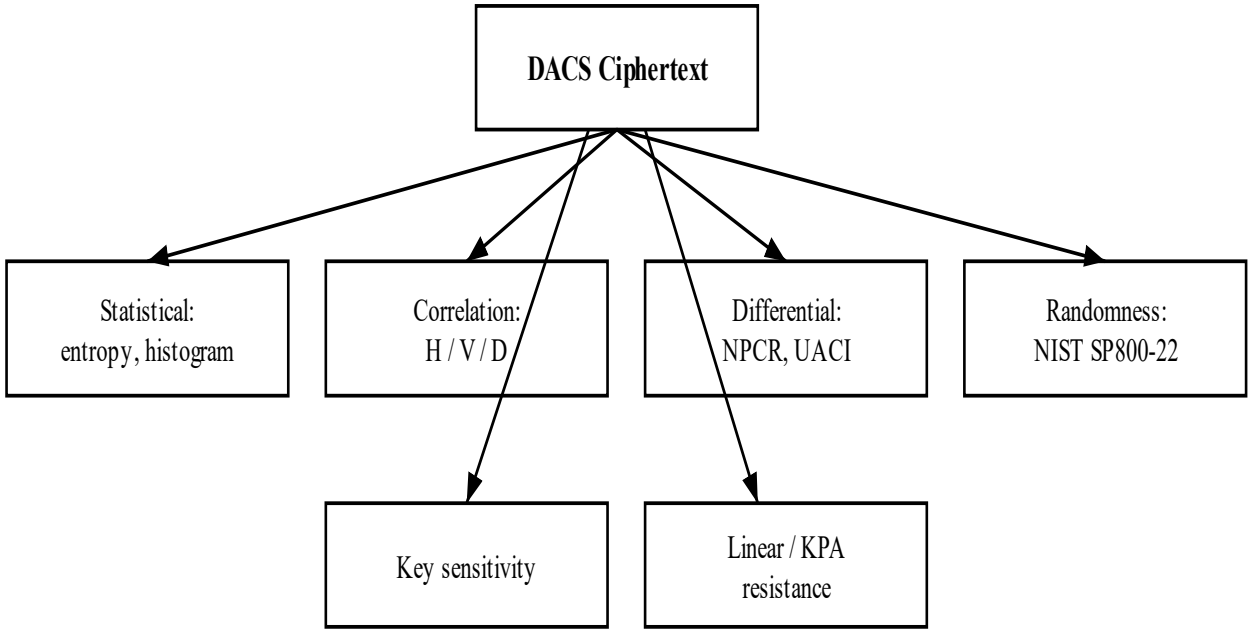

Fig. 11. Security-evaluation framework applied to DACS.

5. Discussion

Three things stand out in the measurements. The first is that DACS is, statistically, indistinguishable from the strongest recent chaos-based ciphers. Its entropy lands within 8×10^{-4} of the 8-bit ceiling, its pixel correlations sit around a thousandth, and its histogram passes the χ^2 uniformity test outright. These are exactly the properties an attacker leans on in a frequency or statistical attack, and DACS leaves them nothing to work with. The reason is structural: the 2D-CLT generator hands the cipher a broadband, near-uniform stream, and the affine–XOR core smears it evenly across the whole byte range. The second point is about what the chaos is actually *doing*. In most designs a chaotic map is there to scramble pixels. Here it does something more pointed—it sets the cipher's own parameters, minting a new (a_i, b_i, k_i) at every position. That is what turns a static linear map into a moving target. The clearest evidence is the attack that did not work: the textbook two-pair known-plaintext attack, the one that reduces an ordinary affine cipher to a pencil-and-paper exercise, predicted subsequent bytes 0.0% of the time. There is simply no single (a, b) left to recover, so the attacker's two equations describe two isolated positions and nothing beyond them. This is the concrete sense in which the affine map's linear weakness has been removed rather than hidden. The third point concerns the plaintext dependency, which is what gives DACS its avalanche. The SHA-256 seed incorporates the message digest, and the state is then chained forward through ciphertext feedback; as a result, flipping even one input bit reshapes the entire downstream keystream—which is where the NPCR and UACI values of 99.6342% and 33.5117%, both just short of the theoretical ideal, come from. The contrast with a classical additive stream cipher is instructive: there the keystream is fixed by the key alone, independent of the message, and reusing it unravels the scheme. In DACS every message receives its own keystream, and that shuts down this entire family of attacks. None of this comes for free, and the honest cost is speed. The reference implementation runs at 28.74 MB/s—about 90.48 cycles per byte—single-threaded and in a high-level language. For a software chaos-based cipher that is a reasonable figure, but it is not in the same conversation as the multi-Gbps numbers reported for dedicated FPGA designs such as [20], which live on different hardware and use fixed-point arithmetic. Two things hold the throughput back: the floating-point chaotic iteration, and the ciphertext-feedback chain, which is sequential by construction and therefore resists the parallelism that would otherwise be the obvious remedy. Both are engineering problems rather than design flaws, and Section 6 sketches how each might be attacked.

6. Conclusion

6.1 Summary of Findings

DACS is a plaintext-dependent dynamic affine–chaotic stream cipher whose affine coefficients are regenerated per byte from a 2D coupled Logistic–Tent system seeded by an SHA-256 schedule and chained through ciphertext feedback. On the standard *cameraman* image it attains a measured entropy of 7.9992 bits, adjacent-pixel correlations of −0.0047 / −0.0053 / 0.0128, a histogram χ^2 of 284.72 (passing at 5%), NPCR of 99.6342%, UACI of 33.5117%, and key sensitivity of 99.5979%. Its key-only keystream passes the evaluated NIST SP 800-22 tests, and the classical static-affine known-plaintext attack succeeds on 0.0% of subsequent bytes. These measured results meet the design goal of an affine-core cipher that is free of the affine map's linear weakness.

6.2 Limitations

The limitations are those observed in the experiments. (i) The reported throughput of 28.74 MB/s is a software, single-thread, floating-point figure and is not competitive with hardware ciphers; the ciphertext-feedback chain prevents straightforward parallelisation. (ii) The randomness evaluation covers eight of the fifteen NIST SP 800-22 tests; the remaining tests are not available in the provided simulation results. (iii) The image-domain evaluation covers three standard grayscale images; colour, video, and general binary formats are not included in the provided data. In addition, the synthetic-IV construction makes encryption deterministic for a repeated (K, IV, P) triple, so per-message IV freshness is required when semantic security across identical messages matters. (iv) The cross-method comparison relies on values reported by other authors under their own conditions and is therefore indicative rather than a controlled benchmark.

6.3 Future Work

Three directions follow from the limitations. A *hardware implementation* with fixed-point chaotic arithmetic and a pipelined feedback path would test whether DACS can approach the throughput of FPGA chaotic ciphers. A *post-quantum adaptation* would study DACS as a keystream generator within larger symmetric constructions whose security margins are revisited under Grover-type speed-ups. Finally, *optimisation of the chaotic map*—for example replacing the 2D-CLT with a higher-dimensional hyperchaotic map or the 2D logistic-adjusted-sine map [16]—may further enlarge the parameter space and improve the per-byte mixing while retaining invertibility.

Funding: This research received no external funding.
Conflicts of Interest: The authors declare no conflict of interest.
ORCID iD: 0000-0003-0723-3096[1], 0000-0002-7439-858X[2]

References

[1] M. Matsui, 1994, "*Linear cryptanalysis method for DES cipher*," in *Advances in Cryptology – EUROCRYPT '93*, LNCS, vol. 765, pp. 386–397. doi:10.1007/3-540-48285-7_33.
[2] G. Alvarez and S. Li, 2006, "*Some basic cryptographic requirements for chaos-based cryptosystems*," International Journal of Bifurcation and Chaos, vol. 16, no. 8, pp. 2129–2151. doi:10.1142/S0218127406015970.
[3] C. E. Shannon, 1949, "*Communication theory of secrecy systems*," Bell System Technical Journal, vol. 28, no. 4, pp. 656–715. doi:10.1002/j.1538-7305.1949.tb00928.x.
[4] M. Alawida, A. Samsudin, J. S. Teh, and R. S. Alkhawaldeh, 2019,"A new hybrid digital chaotic system with applications in image encryption," *Signal Processing*, vol. 160, pp. 45–58, 2019. doi:10.1016/j.sigpro.02.016.
[5] N. Khalil, A. Sarhan, and M. A. M. Alshewimy, 2021,"*An efficient color/grayscale image encryption scheme based on hybrid chaotic maps,*" Optics & Laser Technology, vol. 143, p. 107326,. doi:10.1016/j.optlastec.2021.107326.
[6] S. Kumar and D. Sharma, 2024,"A *chaotic based image encryption scheme using elliptic curve cryptography and genetic algorithm,*" Artificial Intelligence Review, vol. 57,no. 4, pp. 87.
[7] H. Alshehri, 2025,"*Secure image encryption using a 4D chaotic system and Langton's ant cellular automaton*," Scientific Reports, vol. 15, no. 1, pp. 41918. .
[8] H. El Bourakkadi, A. Chemlal, H. Tabti, M. Kattass, and A. Benazzi, 2025,"*A novel image cipher system based on hybrid vigenere-affine approach*," Multimedia Tools and Applications, vol. 84, pp. 34303–34329. doi:10.1007/s11042-024-20532-0.
[9] J. Ahmad and S. O. Hwang, 2015,"*A secure image encryption scheme based on chaotic maps and affine transformation*," Multimedia Tools and Applications, vol. 75, pp. 13951–13976. doi:10.1007/s11042-015-2973-y.

[10] L. Manzoor Ahmad and S. Qureshi *et al.*, 2022,"*RGB image encryption based on symmetric keys using Arnold transform, 3D chaotic map and affine hill cipher*," *Optik*, vol. 266.

[11] S. De, J. Bhaumik, and D. Giri, 2022,"*A new image encryption based on two chaotic maps and affine transform*,", International Conference on Computational Intelligence in Communications and Business Analytics, pp. 100-114.

[12] P. N. Lone, D. Singh, and U. H. Mir, 2021,"*A novel image encryption using random matrix affine cipher and the chaotic maps*," *Journal of Modern Optics*, vol. 68.

[13] Y. Alghamdi, A. Munir, and J. Ahmad, 2022,"*A lightweight image encryption algorithm based on chaotic map and random substitution*," *Entropy*, vol. 24, no. 10, p. 1344. doi:10.3390/e24101344.

[14] S. Mansoor and S. A. Parah, 2023,"*HAIE: a hybrid adaptive image encryption algorithm using chaos and DNA computing*," Multimedia Tools and Applications, vol. 82,no. 19, pp. 28769-28796.

[15] Y. Kaya, Z. Gurkas-Aydin, and A. Akgul, 2025,"*A chaos-based lightweight encryption scheme using hash-code and cyclic rotation*," Physica Scripta, vol. 100, p. 045203.

[16] Z. Hua and Y. Zhou, 2016,"*Image encryption using 2D logistic-adjusted-sine map*," Information Sciences, vol. 339, pp. 237–253.

[17] LE. Bassham III, AL. Rukhin, J. Soto, JR. Nechvatal, ME. Smid, EB Barker, SD. Leigh, M. Levenson, M. Vangel, DL. Banks and NA. Heckert 2010,"*A statistical test suite for random and pseudorandom number generators for cryptographic applications*," NIST Special Publication 800-22 Rev. 1a.

[18] M. A. Ben Farah, R. Guesmi, A. Kachouri, and M. Samet, 2019,"*An image encryption scheme based on a new hybrid chaotic map and optimized substitution box,*" Nonlinear Dynamics. doi:10.1007/s11071-019-05413-8.

[19] A. Abdelli, W. E. H. Youssef, F. Kharroubi, L. Khriji, and M. Machhout, 2023,"*A novel enhanced chaos based present lightweight cipher scheme,*" Physica Scripta, vol. 99, p. 016004.

[20] A. Abdell, W. El Hadj Youssef, L. Khriji and M. Machhout., 2024,"*Enhanced lightweight encryption algorithm based on chaotic systems*," Physica Scripta, vol. 99, no. 10, pp. 106006.

[21] E. Güvenoğlu, 2024,"*An image encryption algorithm based on multi-layered chaotic maps and its security analysis*," Connection Science, vol. 36, no. 1, pp. 1-32.

[22] S.N. Alrekaby, M.A. Khodher, L.K. Adday and R. Aljuaidi, 2025,"*Secure image transmission using multilevel chaotic encryption and video steganography*," Algorithms, vol. 18, no.7, pp. 406.

[23] N. Chaudhary, T. Shahi, and A. Neupane, 2022,"*Secure image encryption using chaotic, hybrid chaotic and block cipher approach*," Journal of Imagin*g*, vol. 8, no. 6.