



Chaos-Based Image Encryption with 5D Keystreams and 3D Voxel Scrambling: A Survey and Reference-Implementation Study

Vittala S¹, Sooraj N J², Vinay B S³, Teja N R⁴, and Anuradha C R⁵

Department of Computer Science and Engineering,

East West College of Engineering, Bengaluru, Karnataka 560064, India,

(Affiliated to Visvesvaraya Technological University, Belagavi)¹⁻⁵

Abstract: Digital images used in medical, financial, surveillance and cloud settings need protection that general-purpose ciphers provide inefficiently, because images are large and strongly redundant. Chaos-based ciphers address this with confusion–diffusion structures driven by keystreams from chaotic and hyperchaotic systems. This paper surveys recent work (2018–2026) on hyperchaotic image encryption, organizes it by chaotic source and cipher structure, and then studies one representative design in depth: 3D voxel scrambling of a 512×512 image combined with a keystream from a five-dimensional Lorenz-type system seeded by SHA-384. A complete Python reference implementation recovers all test images losslessly and gives cipher entropy of 7.9993 bits, adjacent-pixel correlation below 0.02 in magnitude, NPCR of 99.60–99.62 % and UACI of 33.47–33.51 %, all inside the 5 % critical intervals of Wu et al. Two findings are hidden by these headline metrics. First, the Lyapunov spectrum of the 5D system has a single positive exponent (+1.13), so the system is chaotic, not hyperchaotic. Second, the cipher is exactly affine over $\mathbb{Z}_{256}$, which makes its differential behaviour independent of the plaintext and exposes it to chosen-plaintext attack. We conclude with concrete design recommendations and open problems.

Index Terms: image encryption, hyperchaotic system, 3D pixel scrambling, Fisher–Yates shuffle, confusion–diffusion, NPCR, UACI, Lyapunov exponent

I. INTRODUCTION

Images now carry some of the most sensitive data in daily use—radiological scans, banking documents, identity photographs and surveillance footage—and they are routinely stored in the cloud and sent over open networks. General-purpose ciphers such as AES [1] and DES were designed for text-like data [2], [3]. Images are large and highly redundant, so a block cipher applied naively (for example in ECB mode) can leak the outline of the picture, and heavy per-block processing of megabytes of pixels is a poor fit for real-time use.

Chaos-based image encryption has been studied since the late 1990s as an alternative [4], [5]. The usual architecture, introduced by Fridrich [4], alternates a permutation stage that moves pixels (confusion in Shannon’s sense [6]) with a diffusion stage that spreads every pixel’s influence over many others. Chaotic systems are attractive keystream sources because they are deterministic yet extremely sensitive to initial conditions and parameters, which can serve as the secret key. Recent surveys [7], [8] show the field moving toward higher-dimensional and hyperchaotic sources, richer permutations (bit-level, 3D, DNA) and larger evaluation suites.

Problem statement. Images shared online are not completely secure and can be accessed by unauthorized parties; an image-specific encryption system is needed to protect them. *Objectives* of the project behind this paper are to (i) build a secure image encryption system, (ii) apply 3D pixel scrambling to improve confusion, (iii) generate encryption keystreams from a chaotic system, and (iv) perform encryption and decryption with a diffusion stage. *Scope* is still-image encryption for medical, military and cloud use; real-time video is left as future work.

The contributions are: (1) a structured survey of ten recent works on chaotic and hyperchaotic image encryption; (2) a precise description of a reference design—5D keystream plus 3D voxel scrambling—including where the project slides and the code differ; (3) an experimental evaluation on four images against the statistical critical values of Wu et al. [9]; and (4) two analytical findings, on the dynamics of the 5D system and on the algebraic structure of the cipher, with recommendations. Section II gives background, Section III the survey, Section IV the design, Section V the results, Section VI the discussion, and Section VII future directions.



II. BACKGROUND

A. Chaotic Keystreams and the Confusion–Diffusion Structure

A chaos-based cipher derives a keystream by iterating or integrating a chaotic system from a key-dependent state. Sensitive dependence and mixing mirror Shannon's confusion and diffusion [6], and the idea of chaotic encryption dates back at least to Matthews [10]. These properties are necessary but not sufficient: Alvarez and Li [11] list cryptographic requirements that many designs violated, and Fridrich's own scheme was broken by a chosen-plaintext attack [12]. Chen et al. extended the cat map to three dimensions to permute pixels [13], an early example of the 3D scrambling idea used later in this paper.

B. Hyperchaos and Lyapunov Exponents

A dissipative continuous-time system is chaotic if its largest Lyapunov exponent is positive and hyperchaotic if at least two exponents are positive. The Lorenz system [14] is a common base; Jia obtained a hyperchaotic system from it by adding a state [15]. The full spectrum can be estimated by evolving the tangent map alongside the trajectory with periodic QR re-orthonormalization [16], [17]. The Kaplan–Yorke dimension [18] condenses the spectrum,

$$D_{KY} = j + (\lambda_1 + \dots + \lambda_j) / |\lambda_{j+1}|, (1)$$

where j is the largest index for which the partial sum is non-negative.

C. Security and Quality Metrics

Sensitivity to a one-pixel plaintext change is measured on two cipher images C_1 , C_2 with T samples by the number of pixels change rate and the unified average changing intensity [9],

$$NPCR = (100/T) \sum_k [C_1(k) \neq C_2(k)], (2)$$

$$UACI = (100/255T) \sum_k |C_1(k) - C_2(k)|. (3)$$

For uniformly random 8-bit data the expected values are 99.609 % and 33.464 %, and Wu et al. give significance-level critical values that depend on T [9]. We also use Shannon entropy per channel,

$$H = -\sum_{v=0}^{255} p_v \log_2 p_v, (4)$$

the chi-square statistic of the histogram (255 degrees of freedom), correlation of adjacent pixels, and PSNR and SSIM [19] between plain, cipher and decrypted images.

III. LITERATURE SURVEY

A. Surveys and Reviews

Zia et al. [7] classify chaos-based image cryptosystems published in 2016–2021 into spatial, temporal and spatiotemporal domains, compare them with traditional ciphers, and review the evaluation metrics used. Their survey does not converge on one recommended scheme and points to processing time, complexity and attack resistance as continuing issues. Zhang and Liu [8] follow the development of chaotic image encryption across symmetric and asymmetric designs, block and stream structures and integration with other technologies, and discuss applications and open challenges.

B. Hyperchaotic and Higher-Dimensional Schemes

Hyperchaotic sources are favoured because extra positive exponents and state variables give more complex trajectories and larger parameter spaces. Sun [20] seeds a 5D hyperchaotic system to drive pixel-level and bit-level scrambling followed by DNA encoding and XOR. Zhu and Zhu [21] construct a five-dimensional continuous hyperchaotic system and combine it with dynamic DNA coding in a classical scramble–diffuse pipeline. Gao et al. [22] fuse several grayscale images into one colour image and encrypt it with a fractional-order hyperchaotic system; the project review notes the method's high computation and its need for careful parameter selection and synchronization for correct decryption.

Niu et al. [23] improve a 4D chaotic system, generate plaintext-related pseudo-random sequences, and add evolutionary crossover and mutation operators with ciphertext feedback; the reported ciphertext change rate exceeds 0.9967 after a small plaintext change, at the cost of extra processing time and resources. Liu et al. [24] pair a newly designed hyperchaotic system with two-dimensional compressed sensing, so that compression and encryption happen together. Ding et al. [25] use a segmented linear function to build a multi-wing hyperchaotic attractor and encrypt bit-plane-decomposed images; such systems raise cost for large images and real-time use. Subathra and Thanikaiselvan [26] derive initial conditions from statistics of a U-Net-segmented region of a medical image and apply zig-zag scrambling, 5D permutation and dynamic DNA diffusion.

The closest work to the present study is that of Anujaa et al. [27], who modify a Lorenz-derived 5D system, seed it with SHA-512 of the image, and run multi-round row and column shuffling with XOR diffusion plus a secondary image. They report average NPCR 99.6069 %, UACI 33.4284 % and entropy 7.9944 over 100 images, and 0.53 s per image in MATLAB. The base paper of this project, by Riasat and Mehmood [28], combines a 5D hyperchaotic keystream with



3D pixel scrambling and adaptive confusion–diffusion; we could not retrieve its full text, so the design in Section IV follows the project slides and code, not the original equations.

C. Cross-Cutting Observations

Table I summarizes the surveyed works. Four patterns recur. (i) Seeds come from a hash of a key, and in several recent designs of the plaintext as well [23], [27]. (ii) Almost all schemes use a permutation–diffusion order with two or more rounds. (iii) Evaluation converges on entropy, correlation, NPCR/UACI and key sensitivity. (iv) The reported drawbacks are consistent: higher computation cost, parameter sensitivity and difficulty for real-time or large images. What the surveyed papers rarely do is verify the claimed dynamics by computing the full Lyapunov spectrum for the parameter values actually used, or analyse the algebraic structure of the diffusion stage; Section VI shows why both matter.

TABLE I

SUMMARY OF SURVEYED IMAGE-ENCRYPTION WORKS

Ref. (year)	Chaotic core	Technique	Reported results / noted limitation
[7] (2022)	Chaotic maps (survey)	Classification into spatial, temporal and spatiotemporal domains; comparison with traditional ciphers; metric review	No single scheme proposed; processing time, complexity and attacks remain open
[8] (2023)	Chaos (review)	Symmetric/asymmetric, block/stream, integration with other technologies; applications	Survey of challenges; no scheme proposed
[20] (2018)	5D hyperchaotic	Pixel- and bit-level scrambling, DNA encoding, DNA XOR	Illustrates early 5D use in image ciphers
[21] (2020)	New 5D continuous hyperchaotic	Dynamic DNA coding within a scramble–diffuse pipeline	System constructed by the authors
[22] (2021)	Fractional-order hyperchaotic	Multi-image fusion into a colour image, then scrambling and diffusion	High computation; careful parameter choice and synchronization needed
[23] (2024)	Improved 4D chaotic	Plaintext-related sequences; bit/pixel scrambling; crossover and mutation; ciphertext feedback	Change rate > 0.9967 reported; more complexity, time and resources
[24] (2024)	New hyperchaotic system	2D compressed sensing for joint compression and encryption	Passes NIST tests as a generator (reported)
[25] (2025)	Multi-wing hyperchaotic	Segmented linear function; bit-plane decomposition and combination	Complex system; heavy for large or real-time images
[26] (2025)	New 5D hyperchaotic	U-Net region statistics as seeds; zig-zag scramble; dynamic DNA flip/XOR	Targets medical images
[27] (2025)	Modified 5D (Lorenz-derived)	SHA-512 of image as seed; multi-round row/column shuffle with XOR; secondary image	NPCR 99.6069 %, UACI 33.4284 %, H = 7.9944; 0.53 s (MATLAB)
[28] (2026)	5D hyperchaotic	3D pixel scrambling with adaptive confusion–diffusion	Base paper of this project; not independently verified

IV. REFERENCE DESIGN

A. Overview

The design studied here follows the nine-step methodology of the project slides: input image, 2D-to-3D conversion, keystream generation, 3D scrambling, confusion–diffusion, cipher output, decryption, applications and performance evaluation. The implementation has three Python modules: the cipher (chaos_image_cipher.py), the metrics (metrics.py) and a driver (main.py). Fig. 1 shows the data path.

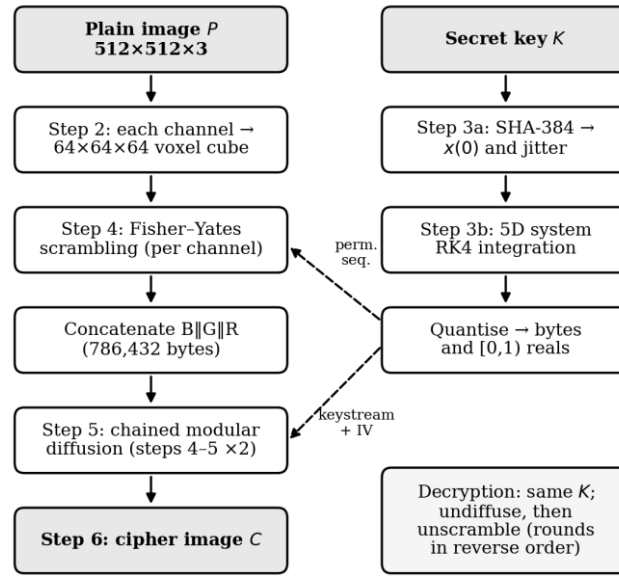


Fig. 1. Encryption and decryption pipeline of the reference design (step numbers follow the project methodology).

B. Key Derivation

The user's secret key string is hashed with SHA-384 [29]. The 48-byte digest is read as twelve big-endian 32-bit words $w_0 \dots w_{11}$, normalized to $u_j = w_j / (2^{32} - 1)$. Words 0–4 set the initial state and words 5–9 perturb the system parameters by at most $\pm 1\%$:

$$x_i(0) = 2u_{i-1} - 1, \quad i = 1, \dots, 5, (5)$$

$$p_j = p_j^0 [1 + 0.02(u_{4+j} - \frac{1}{2})], \quad j = 1, \dots, 5, (6)$$

Independent sequences for different purposes come from appending a label to the key before hashing: perm-rp-{B,G,R} for each channel's permutation, diff-rp for the diffusion keystream and iv-rp for the initial value, with round $\rho \in \{0, 1\}$. One key therefore yields ten seeds.

C. The 5D Chaotic System

The state $(x_1, \dots, x_5)$ evolves by a Lorenz-type system. The variables $x_1 - x_3$ form the classical Lorenz core [14], and x_4, x_5 are damped feedback states, in the spirit of the Lorenz-derived hyperchaotic constructions of Jia [15]:

$$dx_1/dt = a(x_2 - x_1) + x_4, (7)$$

$$dx_2/dt = c x_1 - x_2 - x_1 x_3 + x_5, (8)$$

$$dx_3/dt = x_1 x_2 - b x_3, (9)$$

$$dx_4/dt = -x_2 x_3 + r x_4, (10)$$

$$dx_5/dt = -x_1 x_3 + s x_5, (11)$$

The base parameters are $(a, b, c, r, s) = (10, 8/3, 28, -1, -1)$, perturbed as in Section IV-B. The system is integrated with the fourth-order Runge–Kutta method (RK4) at $\Delta t = 0.005$ after discarding 1,000 transient steps. The README of the project cites a hyperchaotic range for r for this construction; Section V-I tests that claim numerically.

D. Keystream Quantization

Trajectory values are read in row-major order $(x_1, \dots, x_5)$ of step 1, then step 2, and so on). The k -th value is mapped to $[0, 1)$ by its fractional part and to a byte by scaling,

$$s_k = \text{frac}(10^4 |x|), \quad K_k = \lfloor 256 s_k \rfloor. (12)$$

E. 3D Voxel Scrambling

Each 512×512 channel has $262,144 = 64^3$ pixels, so it reshapes exactly into a $64 \times 64 \times 64$ voxel cube. A Fisher–Yates shuffle [30] driven by the chaotic sequence produces a permutation π of the cube's flattened voxel indices: for $i = n-1$ down to 1, swap π_i with π_j , where

$$j = \lfloor s_i(i+1) \rfloor \bmod (i+1). (13)$$

Each channel and each round uses its own permutation. The inverse permutation restores the original order exactly.



F. Chained Modular Diffusion

The three scrambled channels are concatenated (BIGIR, $n = 786,432$ bytes). With keystream K and initial value IV , each cipher byte is chained to the previous cipher byte:

$$C_0 = (P_0 + K_0 + IV) \bmod 256, (14)$$

$$C_k = (P_k + K_k + C_{k-1}) \bmod 256, (15)$$

A change in one plaintext byte therefore propagates to every later byte, across channel boundaries. Because propagation is forward only, the scramble–diffuse pair is run for two rounds with fresh permutations and keystreams; the second scramble moves early-stream changes to late positions of the stream. Additive chaining is used instead of an XOR chain because a single-bit change in an XOR chain stays a constant offset.

G. Decryption, Complexity and Deviations from the Slides

Decryption regenerates the same sequences and runs the rounds in reverse: $P_k = (C_k - K_k - C_{k-1}) \bmod 256$, then the inverse permutations. Time and memory are $O(n)$ apart from the RK4 cost of about 6.3×10^5 steps per encryption; the chained diffusion is inherently sequential. In our analysis the code is treated as the ground truth, because it differs from the slide diagrams in four ways: the diagrams label the diffusion “XOR” while the code adds modulo 256; the data-flow slide derives the permutation by sorting a chaotic sequence while the code uses a chaos-driven Fisher–Yates shuffle; the diagrams show one pass while the code runs two rounds; and arbitrary inputs must first be resized to 512×512 .

TABLE II
IMPLEMENTATION PARAMETERS

Parameter	Value
Image / cube	$512 \times 512 \times 3$; 64^3 voxels per channel
Base parameters (a, b, c, r, s)	(10, 8/3, 28, -1, -1), ± 1 % key jitter
Initial state	$x_i(0) \in (-1, 1)$ from SHA-384
Integrator	RK4, $\Delta t = 0.005$, 1,000-step burn-in
Rounds	$2 \times$ (scramble, diffuse)
Diffusion	Additive chaining mod 256

V. EXPERIMENTAL RESULTS

A. Setup

The project code was run unchanged on a Linux container (single thread) with Python 3, NumPy 2.4.4 [31], OpenCV 4.13.0 [32], scikit-image 0.26.0 [33] and Matplotlib. The slides list a modest platform (Intel Core i3/i5, 4–8 GB RAM, Windows 10/11 or Ubuntu, Python with OpenCV, NumPy and Matplotlib, optional GPU), which is sufficient. Test images are the “astronaut”, “coffee” and “chelsea” samples of scikit-image, resized to 512×512 by area interpolation, and an all-black image as a worst case for statistical leakage. The key is the default of main.py, EWCE-CSE-Phase1-DemoKey; the wrong key changes its last character. Plaintext-sensitivity tests flip the least significant bit of one blue-channel pixel at (0,0), (256,256) or (511,511) and compare all $T = 786,432$ cipher bytes. Correlations use 5,000 random adjacent pairs of the green channel (fixed random seed).

B. Visual and Histogram Analysis

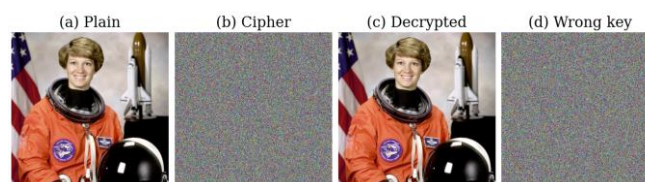


Fig. 2. Astronaut image: (a) plain, (b) cipher, (c) decrypted with the correct key, (d) decrypted with a key differing in one character.

Decryption with the correct key recovered every test image exactly ($PSNR = \infty$, $SSIM = 1$). Fig. 2 shows that the cipher image is noise-like and that a one-character key error gives no visible structure. The cipher histograms in Fig. 3 are flat, and the chi-square statistic of the cipher (Table III) is below the critical value 293.25 (255 d.f., 5 %) for all four images, while it is between 2.98×10^5 and 6.68×10^7 for the plain images.

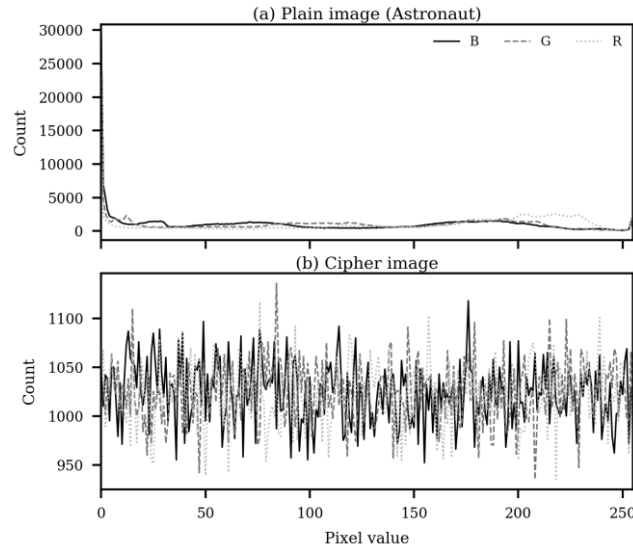


Fig. 3. Pixel-value histograms of the (a) plain and (b) cipher image, per colour channel.

TABLE III
STATISTICAL QUALITY OF THE CIPHER IMAGES

Image	$H(P)$	$H(C)$	$\chi^2(C)$	PSNR (dB)	SSIM
Astronaut	7.3723	7.9993	253.2	7.25	0.0073
Coffee	7.3698	7.9993	259.5	7.43	0.0073
Chelsea	7.0450	7.9994	234.4	9.45	0.0100
Black	0.0000	7.9994	235.0	4.76	4×10^{-6}

H : mean per-channel entropy (bits); χ^2 : mean per-channel statistic, critical value 293.25 (5 %); PSNR and SSIM are between plain and cipher images.

C. Entropy, Correlation and Similarity

The cipher entropy is 7.9993–7.9994 bits per channel against a maximum of 8 (Table III). PSNR between plain and cipher is 4.8–9.4 dB and SSIM is at most 0.010, so the cipher is statistically unrelated to the plain image. Adjacent-pixel correlation, which is 0.96–0.99 in the plain natural images, falls to at most 0.019 in magnitude for the cipher images (Table IV).

TABLE IV
ADJACENT-PIXEL CORRELATION (GREEN CHANNEL)

Image	P: H	P: V	P: D	C: H	C: V	C: D
Astronaut	0.9781	0.9799	0.9667	0.0140	0.0117	0.0018
Coffee	0.9755	0.9841	0.9606	0.0123	0.0107	0.0119
Chelsea	0.9774	0.9863	0.9664	0.0014	0.0042	0.0033
Black	n/a	n/a	n/a	0.0075	0.0109	0.0182

P: plain, C: cipher; H, V, D: horizontal, vertical, diagonal. The black image is constant, so its plain correlation is undefined.

D. Differential and Key-Sensitivity Analysis

At significance 0.05 and $T = 786,432$, the NPCR must be at least 99.598 % and the UACI must lie in [33.411, 33.516] % [9]. All twelve measurements in Table V pass. A one-character key change gives NPCR 99.604 % and UACI 33.464 %, and decryption with that key yields PSNR 7.27 dB and SSIM 0.007 against the original, with entropy 7.9993 bits, i.e. noise. The keystream itself (first-round diffusion sequence, 786,432 bytes) has entropy 7.99978 bits, $\chi^2 = 235.2$ and lag-1 autocorrelation 0.0017.



TABLE V
NPCR AND UACI FOR A ONE-BIT PLAINTEXT CHANGE (%)

Measure / image	(0,0)	(256,256)	(511,511)
NPCR, all four images	99.619	99.603	99.613
UACI, Astronaut	33.468	33.485	33.476
UACI, Coffee	33.506	33.494	33.481
UACI, Chelsea	33.484	33.508	33.478
UACI, Black	33.489	33.492	33.492
Pass criterion (5 %)	NPCR $\geq$ 99.598	UACI in	[33.411, 33.516]

E. Runtime

The first encryption took 13.5 s, about 12 s of which is spent generating the chaotic sequences with a Python loop over small NumPy arrays. Later encryptions and all decryptions in our harness took 1.2–1.4 s because the sequences were cached per (key, length, label); the project's main.py does not cache, so each call pays the full cost. The scheme is therefore practical for still images but far from real time in this implementation, which is consistent with the slides' optional GPU note.

F. Dynamics of the 5D System

We estimated the Lyapunov spectrum of the system in Section IV-C by integrating the variational equation with RK4 ($\Delta t = 0.005$), QR re-orthonormalization every ten steps, a 4,000-step transient and $T = 1,000$ time units [16], [17]. For the base parameters the spectrum is

$$(\lambda_1, \dots, \lambda_5) \approx (1.132, -0.002, -0.993, -2.642, -13.162), (16)$$

with sum -15.67 (dissipative) and $D_{KY} \approx 3.05$. Only λ_1 is positive and $\lambda_2 \approx 0$ corresponds to the flow direction, so the system is *chaotic but not hyperchaotic* (Fig. 4). Scanning $r = s$ over the range quoted in the README (Table VI, $T = 300$) never produced a second exponent clearly separated from zero, and $r = s = 0.2$ and $(r, s) = (-0.5, 0.5)$ diverged. The largest state magnitude observed was about 1.2×10^3 .

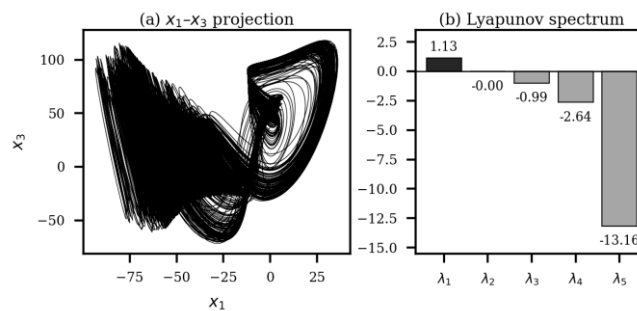


Fig. 4. Base-parameter dynamics: (a) x_1 - x_3 projection; (b) Lyapunov spectrum ($T = 1,000$).

TABLE VI
LYAPUNOV EXPONENTS VERSUS DAMPING $R = S$

$r = s$	λ_1	λ_2	λ_3	λ_4	λ_5
-0.06	0.164	-0.003	-0.266	-1.129	-12.557
-0.1	0.249	0.036	-0.305	-1.231	-12.621
-0.5	0.797	0.009	-0.706	-2.124	-12.645
-1 (used)	1.123	-0.005	-1.015	-2.649	-13.122
-1.5	0.211	-0.002	-0.879	-3.307	-12.690

Estimated with $T = 300$; $|\lambda_2| \leq 0.04$ is within estimation error of zero.



VI. DISCUSSION

A. What the Metrics Do and Do Not Show

On the standard checklist the scheme performs as well as the works in Table I: entropy, correlation, NPCR and UACI are at their random-data values (Table VII). But three properties of the design explain why such a checklist is not enough.

TABLE VII
COMPARISON WITH A RECENT 5D SCHEME

Metric	Ideal	This study	Anujaa [27]
NPCR (%)	99.609	99.612	99.6069
UACI (%)	33.464	33.488	33.4284
Cipher entropy	8	7.9993	7.9944
Encryption time	—	1.2–13.5 s (Python)	0.53 s (MATLAB)

This study: mean of twelve measurements (NPCR: three positions, identical for all images) and four images; test sets and platforms differ, so the rows are indicative only.

B. Affine Structure and Chosen-Plaintext Exposure

Both permutations and the additive chained diffusion are affine over $\mathbb{Z}_{256}$, so the whole cipher is

$$E(P) \equiv L P + E(0) \pmod{256}, (17)$$

where the matrix L and the vector $E(0)$ depend only on the key. We verified this on the implementation: for the astronaut image A , the coffee image B and their pixel-wise sum modulo 256, $E(A) + E(B) - 2E(0)$ equals $E(A+B) - E(0)$ at all 786,432 positions (zero mismatches), and the cipher difference caused by a one-LSB change is identical for both images. This has two consequences. First, the NPCR values in Table V are identical for all four images: NPCR here measures a fixed property of L , not sensitivity to the plaintext. Second, an adversary with encryption-oracle access under a reused key can recover $E(0)$ and, with at most $n = 786,432$ further single-pixel queries, all columns of L , and then decrypt any cipher image without the key. We did not mount this attack, but it follows directly from the affine structure and matches the chosen-plaintext weaknesses documented for classical permutation–diffusion designs [12], [11]. Plaintext-dependent seeds, as in [23] and [27], remove the fixed map, provided the plaintext-derived value is protected.

C. Other Design Observations

The 3D cube is bookkeeping. The permutation acts on the flattened voxel index, so the $64 \times 64 \times 64$ view gives the same cipher as permuting the 2D channel directly; a real 3D advantage would need permutations that use the three axes, as in 3D cat maps [13]. *Hyperchaos is not achieved.* With one positive exponent (Section V-F), the title, slides and README should say “5D chaotic (Lorenz-type)”. The keystream statistics are nonetheless good, so this is a claim-accuracy issue, not a randomness failure. *Key derivation.* SHA-384 uses 10 of 12 words, i.e. 320 bits, but a passphrase carries far less entropy, and a bare hash offers no protection against dictionary attack; a password-based KDF, a password-based KDF [34] should be used. *Portability.* With $\lambda_1 \approx 1.13$ and up to about 2.6×10^2 time units per sequence, a one-ulp difference grows by roughly e^{296} ; decryption therefore needs bit-identical floating-point arithmetic on both sides, which fixed-point or integer chaotic maps avoid.

D. Recommendations

(1) Mix a hash of the plaintext or a per-image nonce into the seeds. (2) Break the affine structure with a nonlinear step, such as an S-box, XOR combined with modular addition, or bit-level operations. (3) Report the full Lyapunov spectrum for the parameters actually used, or choose a system with two verified positive exponents. (4) Add NIST SP 800-22 tests on the keystream and noise and cropping robustness tests. (5) Use a key-stretching KDF. (6) Move the RK4 loop to compiled or GPU code and support arbitrary image sizes with padding.

VII. FUTURE DIRECTIONS

The project scope names real-time video encryption; the chained diffusion would need to be replaced by a parallelizable one, and inter-frame redundancy considered. Joint compression and encryption [24], region-of-interest encryption for medical images [26], FPGA implementation of the chaotic generator, and cryptanalysis-driven design and shared benchmarks that include chosen-plaintext resistance are further directions.

VIII. CONCLUSION

This paper surveyed recent chaotic and hyperchaotic image encryption and analysed one representative design—3D voxel scrambling with a SHA-384-seeded 5D keystream—through a working implementation. The design decrypts losslessly and meets the standard statistical criteria on four test images. Closer analysis, however, shows that its 5D source is chaotic rather than hyperchaotic, that the 3D cube adds nothing beyond a 1D permutation, and that the cipher is affine over $\mathbb{Z}_{256}$, so its high NPCR and UACI do not imply resistance to chosen-plaintext attack. Plaintext-dependent



seeding and a nonlinear diffusion step are the most direct repairs. We hope the survey table and the analysis method are useful for evaluating similar schemes.

ACKNOWLEDGMENT

The authors thank the Department of Computer Science and Engineering, East West College of Engineering, for supporting this project (Phase-1 review).

REFERENCES

- [1] NIST, "Advanced Encryption Standard (AES)," Federal Inf. Process. Stand. Publ. FIPS 197, Nov. 2001.
- [2] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 7th ed. Harlow, U.K.: Pearson Education, 2017.
- [3] D. R. Stinson, *Cryptography: Theory and Practice*, 3rd ed. Boca Raton, FL, USA: CRC Press, 2005.
- [4] J. Fridrich, "Symmetric ciphers based on two-dimensional chaotic maps," *Int. J. Bifurcation Chaos*, vol. 8, no. 6, pp. 1259–1284, 1998.
- [5] L. Kocarev, "Chaos-based cryptography: A brief overview," *IEEE Circuits Syst. Mag.*, vol. 1, no. 3, pp. 6–21, 3rd Quart., 2001.
- [6] C. E. Shannon, "Communication theory of secrecy systems," *Bell Syst. Tech. J.*, vol. 28, no. 4, pp. 656–715, Oct. 1949.
- [7] U. Zia, M. McCartney, B. Scotney, J. Martinez, M. AbuTair, J. Memon, and A. Sajjad, "Survey on image encryption techniques using chaotic maps in spatial, transform and spatiotemporal domains," *Int. J. Inf. Secur.*, vol. 21, no. 4, pp. 917–935, 2022.
- [8] B. Zhang and L. Liu, "Chaos-based image encryption: Review, application, and challenges," *Mathematics*, vol. 11, no. 11, Art. no. 2585, 2023.
- [9] Y. Wu, J. P. Noonan, and S. Agaian, "NPCR and UACI randomness tests for image encryption," *Cyber J.: Multidiscip. J. Sci. Technol., J. Sel. Areas Telecommun.*, pp. 31–38, Apr. 2011.
- [10] R. Matthews, "On the derivation of a 'chaotic' encryption algorithm," *Cryptologia*, vol. 13, no. 1, pp. 29–42, 1989.
- [11] G. Alvarez and S. Li, "Some basic cryptographic requirements for chaos-based cryptosystems," *Int. J. Bifurcation Chaos*, vol. 16, no. 8, pp. 2129–2151, 2006.
- [12] E. Solak, R. Çokal, O. T. Yildiz, and T. Biyikoglu, "Cryptanalysis of Fridrich's chaotic image encryption," *Int. J. Bifurcation Chaos*, vol. 20, no. 5, pp. 1405–1413, 2010.
- [13] G. Chen, Y. Mao, and C. K. Chui, "A symmetric image encryption scheme based on 3D chaotic cat maps," *Chaos Solitons Fractals*, vol. 21, no. 3, pp. 749–761, 2004.
- [14] E. N. Lorenz, "Deterministic nonperiodic flow," *J. Atmos. Sci.*, vol. 20, no. 2, pp. 130–141, 1963.
- [15] Q. Jia, "Hyperchaos generated from the Lorenz chaotic system and its control," *Phys. Lett. A*, vol. 366, no. 3, pp. 217–222, 2007.
- [16] G. Benettin, L. Galgani, A. Giorgilli, and J.-M. Strelcyn, "Lyapunov characteristic exponents for smooth dynamical systems and for Hamiltonian systems; a method for computing all of them," *Meccanica*, vol. 15, pp. 9–20, 1980.
- [17] A. Wolf, J. B. Swift, H. L. Swinney, and J. A. Vastano, "Determining Lyapunov exponents from a time series," *Physica D*, vol. 16, no. 3, pp. 285–317, 1985.
- [18] J. L. Kaplan and J. A. Yorke, "Chaotic behavior of multidimensional difference equations," in *Functional Differential Equations and Approximation of Fixed Points*, Lecture Notes in Math. vol. 730. Berlin, Germany: Springer, 1979, pp. 204–227.
- [19] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
- [20] S. Sun, "A novel hyperchaotic image encryption scheme based on DNA encoding, pixel-level scrambling and bit-level scrambling," *IEEE Photon. J.*, vol. 10, no. 2, pp. 1–14, Apr. 2018.
- [21] S. Zhu and C. Zhu, "Secure image encryption algorithm based on hyperchaos and dynamic DNA coding," *Entropy*, vol. 22, no. 7, Art. no. 772, 2020.
- [22] X. Gao, J. Yu, S. Banerjee, H. Yan, and J. Mou, "A new image encryption scheme based on fractional-order hyperchaotic system and multiple image fusion," *Sci. Rep.*, vol. 11, 2021, doi: 10.1038/s41598-021-94748-7.
- [23] Y. Niu, H. Zhou, and X. Zhang, "Image encryption scheme based on improved four-dimensional chaotic system and evolutionary operators," *Sci. Rep.*, vol. 14, 2024, doi: 10.1038/s41598-024-57756-x.
- [24] M. Liu, C. Ning, and C. Zhu, "A secure image encryption scheme based on a new hyperchaotic system and 2D compressed sensing," *Entropy*, vol. 26, no. 7, Art. no. 603, 2024.
- [25] P. Ding, W. Hu, P. Geng, J. Zhang, and J. Zhu, "A new multi-wing hyperchaotic system and its application in image encryption," *J. Supercomput.*, vol. 81, Art. no. 1201, 2025.
- [26] S. Subathra and V. Thanikaiselvan, "Enhanced security for medical images using a new 5D hyper chaotic map and deep learning based segmentation," *Sci. Rep.*, vol. 15, Art. no. 22628, 2025.



- [27]T. Anujaa *et al.*, “A lightweight multi round confusion-diffusion cryptosystem for securing images using a modified 5D chaotic system,” *Sci. Rep.*, vol. 15, Art. no. 31986, 2025.
- [28]S. Riasat and E. Mehmood, “Secure image encryption using 5D hyper-chaotic keystream and 3D pixel scrambling with adaptive confusion-diffusion,” in *Proc. IEEE ICACS*, 2026 (as cited in the project review; full text not independently retrieved).
- [29]NIST, “Secure Hash Standard (SHS),” Federal Inf. Process. Stand. Publ. FIPS 180-4, Aug. 2015.
- [30]R. Durstenfeld, “Algorithm 235: Random permutation,” *Commun. ACM*, vol. 7, no. 7, p. 420, Jul. 1964.
- [31]C. R. Harris *et al.*, “Array programming with NumPy,” *Nature*, vol. 585, pp. 357–362, 2020.
- [32]G. Bradski, “The OpenCV Library,” *Dr. Dobbs’s J. Softw. Tools*, 2000.
- [33]S. van der Walt *et al.*, “scikit-image: Image processing in Python,” *PeerJ*, vol. 2, Art. no. e453, 2014.
- [34]B. Kaliski, “PKCS #5: Password-based cryptography specification version 2.0,” IETF RFC 2898, Sep. 2000.